Automated analysis of product related data generated from PLM ontology

THÈSE N^{O} 6728 (2015)

PRÉSENTÉE LE 25 SEPTEMBRE 2015

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DES OUTILS INFORMATIQUES POUR LA CONCEPTION ET LA PRODUCTION
PROGRAMME DOCTORAL EN SYSTÈMES DE PRODUCTION ET ROBOTIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Ana MILICIC

acceptée sur proposition du jury:

Dr A. Karimi, président du jury
Prof. D. Kyritsis, directeur de thèse
Prof. R. Anderl, rapporteur
Prof. M. Garetti, rapporteur
Prof. D. Kuhn, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE
Suisse
2015

Acknowledgements

I would like to express my deepest gratitude to my supervisor Dr. Dimitris Kiritsis for accepting me as his PhD student and for guiding me through the four years of the thesis. I would like to thank him for providing me with the freedom to explore different directions and for his great support through this process.

I wish to thank Prof. Alireza Karimi for his accepting to be the jury president for my PhD oral exam and making the exam process comfortable for me. I am thankful to Prof. Daniel Kuhn, Prof. Reiner Anderl and Prof. Marco Garetti for accepting to participate in my thesis jury and their valuable suggestions.

Also I would like to thank to all my colleagues from LinkedDesign project, for amazing experience of working together and learning from each other. Furthermore, I would like to thank my colleagues from LICP and DK team for friendly and comfortable atmosphere at EPFL, day after day. Special thanks goes to Soumaya for being tolerant and willing to invest her time into supporting me.

I want to thank my friends in Lausanne, for sharing the PhD program experience and going together through the best and the worst of it. Thank you for never letting me feel alone in the face of challenges. Another thanks goes to my friends in Novi Sad, for standing by me despite the distance and reminding me of who I am outside the academia.

Finally, but the most importantly I want to thank my sister Jelena, for being smart and rational, when I wasn't, my mother Marija for being trusty comfort and support and my father Milan for teaching me that there should be no limits in what you strive for.

Abstract

In recent years, ontology for the Product Lifecycle Management (PLM) domain has raised a lot of interest in research communities, both academic and industrial. It has emerged as a convenient method for supporting the concept of closed lifecycle information loop (Jun, Kiritsis, & Xirouchakis, 2007), which is one of the most important issues of PLM. By modeling relevant aspects collected from all lifecycle stages of a product, within one ontology, a common knowledge structure is created accessible to all actors. Assuming that appropriate mechanisms for updating ontology (or rather, instances that populate it) are provided, ontology becomes a base layer for a knowledge management platform. Useful experience and information from Middle of Life (MOL) and End of Life (EOL) stages, can influence designer's decisions and business strategies. The industrial research community has recognized this added value of ontological implementation, and there is an increasing number of developed ontologies for this purpose (Ana Milicic, Perdikakis, El Kadiri, & Kiritsis, 2013). Application of ontology contributes to time efficiency by reducing the time required to retrieve information. Furthermore, it allows for the enhancement of design decisions which are supported through additional information at the appropriate moment. Finally, ontology gives an overall perspective on a product's lifecycle, allowing from-the-top optimization. On the other hand, ontology is a semantic model making it appealing to the research community working on new internet technology, especially Internet of Things (IoT) (Kiritsis, 2011). Different domains modeled in ontology, and software platforms that use them as a base layer, become interoperable and convenient to merge. Trends are developing in such a manner as to make ontology a building block for integrating heterogonous environments that are encountered during the development of complex smart digital environments.

The purpose of ontology as it is today is not to store data, for the most part because there are more efficient data base systems to handle large data amounts. Still, the domain modeled within ontology is composed of structured and un-structured data sets, and ontology itself can give us a top view on relations and dependences between these data sets. In this perspective, it holds a strong similarity to a relational data base, if relations in the data base where defined so that they depicted the real world in the most precise possible manner.

In large companies today, handling a growing amount of data generated every day is becoming an increasingly relevant problem. Managing and storing them, although challenging, is still feasible, but holding data without understanding it carries little added value. In an effort

to exploit useful information contained in unstructured data sources, a number of decision support systems and enterprise resource planning systems have been developed. They can be very diverse in functionality and efficiency but the one thing that they all have in common is that the user has to be the one making the initiative and defining the queries. This means that the user has to know which information he is looking for, or hoping to extract. As a consequence, the number of relevant correlations and dependencies between different factors of real life captured in the data are left unnoticed, simply because they were not assumed. In the PLM domain, this is particularly present since it involves a number of actors and most of them are interacting only with a small subset of domain concepts. Data mining as a discipline, gives a number of tools for resolving this issue. All of the algorithms are designed to detect correlations, underlying patterns or functions that generated the data. The problem of data mining techniques is that they are still performed mostly manually. Although deterministic steps of data mining procedures can be supported by existing software tools, the others remain an obstacle and they necessarily require human-expert involvement. Without going into details, the two main challenges of data mining automation are:

(1) Creating a data set composed out of relevant attributes and appropriate target variables, which is free of noisy data, missing values and other undesirable properties;
(2) Interpreting the results in terms of accuracy measurement selection and algorithm selection accordingly.

This research resulted in a merger of two complementary fields, where the gaps and downfalls of the one were substituted with the benefits of the other. By merging knowledge management through PLM ontology on one side, and data mining, exploiting computer computational power on the other side, we defined an autonomous system for pattern recognition and knowledge detection in the PLM domain, which can be used as a "black box", by non-experts in the data mining field.

Keywords:

PLM, PLM Data, Ontology, Ontology design, Ontology Reasoning, Data Mining

Résumé

Dans les dernières années, les ontologies pour le domaine de la gestion du cycle de vie des produits (GCP), ont suscité beaucoup d'intérêt au sein des communautés de recherche, à la fois académiques et industrielles. Il a émergé comme une méthode pratique pour soutenir le concept de l'information de cycle de vie en boucle fermée (juin, Kiritsis, et Xirouchakis, 2007), qui est l'un des aspects les plus importants de GCP. En modélisant les aspects pertinents recueillies lors de toutes les étapes du cycle de vie d'un produit, en une seule ontologie, une structure commune de connaissances est créé accessible à tous les acteurs. En supposant que des mécanismes appropriés pour mettre à jour l'ontologie (ou plutôt, les instances qui la peuplent) sont fournis, l'ontologie devient une couche de base de la plate-forme de gestion des connaissances. Des expériences et des informations utiles de la phase d'utilisation (MOL) et la phase de fin de vie (EOL), peuvent influencer les décisions des concepteurs et les stratégies commerciales. La communauté de la recherche dans le domaine industriel a reconnu la valeur ajoutée de la mise en œuvre des ontologies, et il y a un nombre croissant des ontologies développées à cet effet (Ana Milicic, Perdikakis, El Kadiri, et Kiritsis, 2013). L'application des ontologies contribue à améliorer l'efficacité en temps en réduisant le temps nécessaire pour récupérer les informations. En outre, il permet l'amélioration des décisions en phase de conception qui sont soutenus par des informations supplémentaires au moment approprié. Enfin, l'ontologie donne une perspective globale du cycle de vie des produits, permettant une « from-the –top » optimisation. D'autre part, l'ontologie est un modèle sémantique qui est attrayant pour la communauté des chercheurs dans le domaine des nouvelles technologies technologie de l'Internet et plus particulièrement l'internet des objets (IdO) (Kiritsis, 2011). Différents domaines modélisés dans les plates-formes d'ontologies et de logiciels qui les utilisent comme couche de base deviennent interopérables et appropriés pour être fusionné. Des tendances se développent de telle façon que l'ontologie sera un bloc de construction pour l'intégration d'environnements hétérogènes qui sont rencontrées lors de développement d'environnements numériques intelligents et complexes.

Le but d'une ontologie tel qu'il est perçu aujourd'hui, est de ne pas stocker les données, surtout parce qu'il y a des systèmes de base de données plus efficaces pour traiter les grandes quantités de données. Pourtant, le domaine modélisé dans l'ontologie comprend ensembles de données structurés et non structurés, et l'ontologie elle-même peut nous donner une vue de dessus sur les relations et dépendances entre ces ensembles de données.

Dans cette perspective, elle détient une forte similitude avec une base de données relationnelle, mais les relations dans la base de données dont définis de façon à ce qu'elles représentent le monde réel de manière la plus précise possible.

Dans les grandes entreprises aujourd'hui, la manipulation d'une quantité croissante de données générées chaque jour, devient un problème de plus en plus pertinent. Les gérer et les stocker, bien que cela représente un défi, reste encore faisable, mais maintenir des données sans les comprendre a peu de valeur ajoutée. Dans l'effort d'exploiter les informations utiles contenues dans les sources de données non structurées, de nombreux systèmes d'aide à la décision et de planification des ressources d'entreprise ont été développés. Ils peuvent être très différents dans la fonctionnalité et l'efficacité, mais une chose qu'ils ont tous en commun est que l'utilisateur doit être celui qui prend l'initiative et définit les requêtes. Cela signifie que l'utilisateur doit savoir quelles informations rechercher ou souhaiter extraire. Par conséquent, les corrélations et dépendances pertinentes entre les différents facteurs de la vie réelle capturées dans les données, restent inaperçues, tout simplement parce qu'ils ne sont pas supposées. Dans le domaine GCP, ceci est particulièrement présent car il implique de nombreux acteurs et la plupart ne sont en interaction qu'avec un petit sous-ensemble du domaine des concepts. L'exploration de données en tant que discipline, fournit un certain nombre d'outils pour résoudre ce problème. Tous les algorithmes sont conçus pour détecter des corrélations, soulignant les motifs ou les fonctions qui ont généré les données. Le problème des techniques d'exploration de données est qu'elles sont encore effectuées le plus souvent manuellement. Bien que des étapes déterministes des procédures d'exploration de données puissent être soutenues par des outils logiciels existants, les autres restent un obstacle et nécessitent une intervention des experts humains. Sans entrer dans les détails, deux principaux défis de l'automatisation de l'extraction de données sont:

(1) Création de l'ensemble de données composé d'attributs pertinents et variables cibles appropriées, qui est exempt de données bruitées, valeurs manquantes et d'autres propriétés indésirables
(2) L'interprétation des résultats dans un sens de la sélection de mesure de la précision et de la sélection de l'algorithme en conséquence

Cette recherche a abouti à la fusion de deux domaines complémentaires, où les lacunes et les chutes d'un sont substituées par les avantages de l'autre. Par la fusion de la gestion des connaissances grâce à une ontologie GCP ontologie et l'extraction de données d'un côté et l'exploitation de la puissance de calcul de l'ordinateur de l'autre côté, nous avons défini un système autonome pour la reconnaissance des formes et de détection de la connaissance dans le domaine PLM, qui peut être utilisé comme une ''boîte noire'' par des non experts dans le domaine de l'exploration de données.

Mots-clés:
GCP, données GCP, ontologie, conception d'ontologie, raisonnement ontologique, exploration de données

Contents

_Toc421782781

List of Figures

List of Tables

List of Equations

List of Acronymes

BOL	Beginning Of Life
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CAM	Computer Aided Manufacturing
CSV	Comma Separated Value
DME	Data Mining Engine
EDM	Engineering Data Management
EOL	End Of Life
ERP	Enterprise Resource Planning
FOL	First Order Logic
IoT	Internet of Things
KBE	Knowledge Based Engineering
KM	Knowledge Management
KNN	K-Nearest Neighbor
LCC	Life Cycle Cost
MOL	Middle Of Life
MSE	Mean Squared Error
ORSD	Ontology Requirements Specification Document
PDM	Product Data Management
PLM	Product Lifecycle Management
PPM	Project Portfolio Management
RBF	Radial Basis Function
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RIF	Rule Interchange Format
SVM	Support Vector Machine
SWRL	Semantic Web Rule Language
TDM	Technical Data Management
USM	User Story Mapping

Chapter 1 Introduction

Recently, using ontology to model product related data is becoming an increasingly accepted trend. Advantages gained are the sharing of knowledge between different participants, defined vocabularies, reasoning capabilities and many more. This is a vital tool for business-implementation as it allows us to model the entire life-cycle of a product, with all its details and dynamics, leading it from one phase to a sequent one. Once the model is created, ontology, since being machine understandable, will allow automatic reasoning and derivation of new rules, describing dependencies and correlations between different factors in a product's life-cycle, extending on the model and knowledge of a product. Although this method for knowledge extraction is irreplaceable, some downfalls have been noticed. First, new knowledge can only be implied from already existing rules, meaning that knowledge which can't be anticipated (which is not a direct consequence of already existing knowledge), will never be extracted. Second, these rules are strictly deterministic, while it is highly expected that humans could benefit from the knowledge in a form of probabilistic correlations and expectations. Both of these issues can be tackled using well-known tools of statistical analysis and data-mining modeling. Numerical modeling and analysis of data is an efficient manner for discovering underlying co-dependences between different factors in a product's life-cycle, as well as giving probabilistic expectations for the behavior of these factors. On the other hand, pattern recognition and data analysis are still done manually which consume a lot of time and effort from human experts. One of the advantages of ontology as a tool for structuring data is that it's both, human and machine understandable, meaning, machines are given the possibility to infer relations between concepts, recognize synonyms and query concept attributes across an entire domain. Such an advantage opens up an entire new field of possibilities for automating data analysis and data-mining modeling in PLM, as a large portion of required steps and operations could be initiated by machines without human engagement.

1.1 Motivation

Every day, production, exploitation and recycling of different products generates large amounts of data which contain vital information and correlations, useful for the further de-

velopment and optimization of a product. Still, statistical analysis of this data is usually performed by human experts, familiar with specific domains, who have to be trained to know in advance what the dependences, relevant and worth looking for, are. Recent trends for implementing ontology as a system for product-related data structuring opens up a number of possibilities. Semantic structures bring advantages such as visibility, advanced querying, interoperability between different systems and many more, thus vast amounts of research efforts have been directed towards methods and tools for efficient data structuring. Yet, only recently, the idea of drawing data sets from complex semantic data structures has emerged as a solution for the quick generation of data sets for numerical analysis tasks. In a sense, this step closes the data handling loop. The process starts with gathering unstructured data from all relevant sources and possibly unrelated actors, in order to create unified ontological structure which are a credible model of reality. This model is the centerpiece for all actors, it enables efficient communication, exchange and re-usage of knowledge and can adopt a number of functionalities in addition to which it gives insight into what factors from the domain are actually dependent on reality and thus worth exploring for patterns in a numerical sense. With this motivation, the data is again brought to a somewhat unstructured, tabular form which is convenient for data mining. Tabular data is an expression used in Excel data sheet terminology describing matrices of data where each row is one instance or data point, while the columns are attributes, properties or dimensions of that instance. The result of data mining in tabular form is again in the form of knowledge that can be built back in ontology, closing the loop. In particular, this has great potential within the domain of Product Lifecycle Management (PLM) where the actors are numerous, often without direct communication or data exchange, and where the dependencies between factors can easily be overlooked while on the other hand hold great benefits in a sense of time efficiency, quality control, safety and overall productivity.

With this concept in mind, the first step of this research is focused on designing template ontology that can be easily applied to most specific PLM domains with a focus on ontological exploitation through data analysis and ontological reasoning. Ontological rules are examined as a method for gaining additional knowledge and enriching ontological structure to bring a better understanding of dependent factors. Finally, the data mining engine for PLM data is designed, so that it can detect patterns and conclusions in a fully automated manner. It can be seen as a "black box" for data mining that can handle any specific task coming from PLM related data without any human involvement.

1.2 Research questions

The process of modeling ontology can be extremely time and effort consuming, especially for complex PLM domains. On the other hand, the more complex the domain is, the more obvious the benefits of exploiting the ontology are. For this reason, the first research question is about ontology design and how to simplify it. Following ontological design, the benefits of ontology need to be examined, with a focus on how to exploit ontological reasoning in order to increase knowledge captured in the domain. This step defines an added value of using ontology over some other semantic model. After having populated the ontological structure, the next step is to extract tabular data. Consequently, the next research question is how to determine which properties and factors are potentially correlated or causal in order to include them in a tabular data set. Once the data is extracted, the next step is to examine statistical dependences between all factors and determine what constitutes a relevant finding. This imposes a research question about defining the threshold for result relevancy. An additional step is the detection of outliers. While in traditional data analysis, outliers are undesirable and usually a consequence of errors and noise in the data, in the PLM domain, these data points can carry useful information about equipment malfunctioning or human error. After applying statistical analysis, pattern recognition is employed in order to detect a more complex and predictive numerical model. Pattern recognition procedures denote exploiting learning algorithms, meaning algorithms which will result in a function or a mathematical model that generates target variables based on given input, with as-high-as possible accuracy. Open questions in this step are how to choose which learning algorithm should be applied, how to compare its performance and how to estimate the quality of the resulting model. Finally, considering that the data mining engine is designed to be used by everyone, including non-experts, the last question is about representation of results. The described process and information flow are represented in Figure 1.

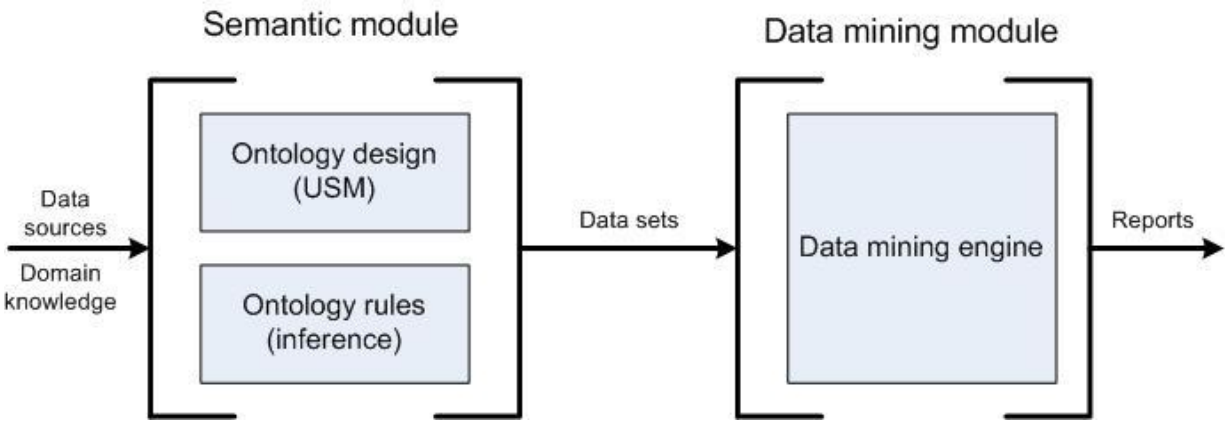


Figure 1 Contribution structure

Summarizing the research questions :

(1) How to simplify PLM ontology modeling for data sources integration?
(2) How to exploit ontology reasoning in PLM domain?
(3) What is the optimal architecture of data mining engine considering data in PLM?
(4) How to extract tabular data from ontology?
(5) What is the threshold for statistical significance of correlations between properties?
(6) How to select optimal learning algorithm?
(7) How to present or visualize results to an end-user?

Research question 1 is answered by applying a method from software design, called User Story Mapping (USM), to domain exploration this results in upper ontology that can be used as a template for any new project in the domain of PLM. Ontology reasoning benefits are shown through the design of a number of unconventional functionalities using rule chaining. Data extraction is solved by defining the graph distance bound, up to which the concepts are considered as related. Research questions 3, 4 and 5 are answered by addressing relevant literature, with consideration for the nature of the data. This results in an automated data processing system that is domain independent and can handle a variety of data characteristics. Finally, through use-cases, end-user feedback was examined and the final choice, for result presentation to an end user, was defined.

1.3 Methodology

The methodology followed in this Dissertation is illustrated in Figure 2. It consists of five successive phases describing the study of background works and technologies, the development of new concepts and methods, the evaluation of the developed ontology, the implementation and testing of developed concepts and methods in case studies, and possible future extensions on this work.

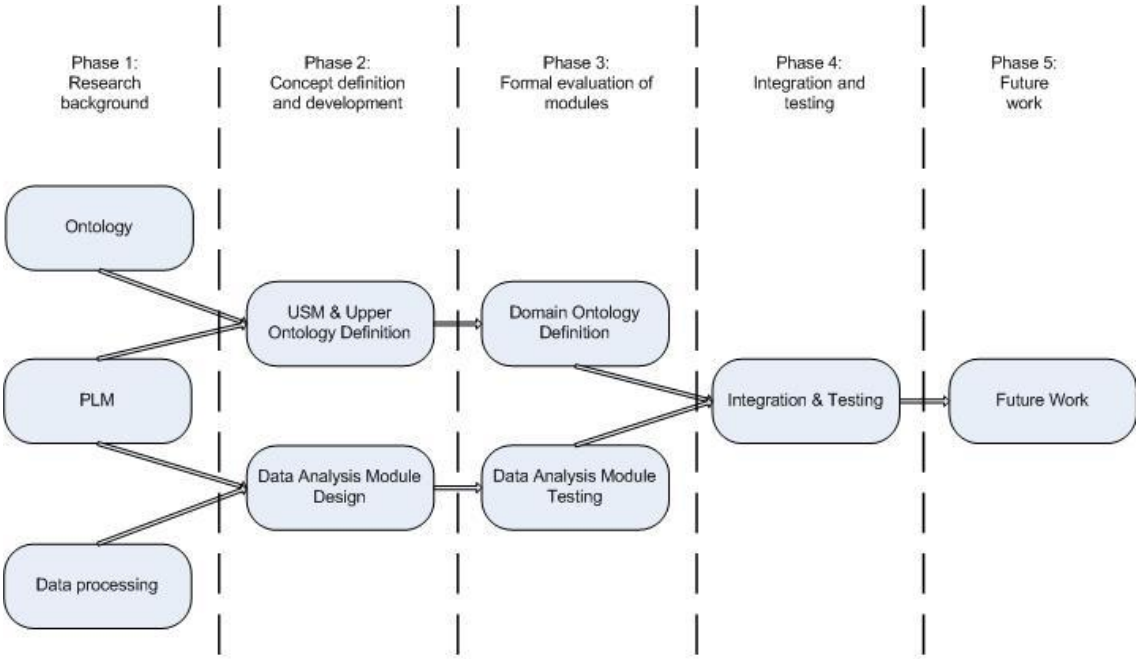


Figure 2 Research methodology

Considering that this Dissertation is an interdisciplinary project, extensive research in three different fields was conducted in Phase 1. The aim was to acquire knowledge on ontological concepts, ontological application, ontological reasoning and ontological design strategies. Following was an introduction to the PLM domain, the main factors and main issues when handling knowledge in closed-loop life-cycle of a product. Finally, extensive research was done on existing tools for data processing, as well as challenges of automation in this process.

In the Phase 2, the first integration of disciplines was conducted. Namely, how to exploit the specificity of the PLM domain and its data sources to simplify ontological design process was examined. This resulted in the application of a USM method and design of upper ontology, spanning the entire PLM domain on an abstract level. Such ontology can be used as a template or guideline for the design of more specific domain ontologies. Further on, how the ontology reasoning mechanism can be employed to create added functionalities of ontology, required by PLM's specific needs, was examined. The second interdisciplinary product came from understanding the type and nature of data used in PLM, as well as the complexity of expected patterns. The result were assumptions that the patterns to be detected are respectively simple in nature, that the outliers carry useful information, more often than being noise, and that high precision of detected patterns is secondary to automation of detecting patterns. The data mining module was designed accordingly.

In Phase 3, the testing of both modules from the previous phase was done independently. In order to test the semantic module and upper ontology template applicability, three specific

domains where used, all required by industrial partners in European FP7 LinkedDesign project. The first domain is semantic enrichment of industrial standards where an extract from the Norsok standard for oil drilling platform design was analyzed and modeled. The second domain is product life-cycle cost modeling together with end user requirements which influence different costs of a product. The third example, the maintenance of a knowledge base was modeled, where knowledge units K-briefs maturity is examined through relevant factors. The data mining module was tested using publicly available data sets for which the highest achievable accuracy is known. By using these data sets, it was determined that the designed, fully automated engine is either as-good-as or slightly less precise than the best, manually tuned algorithms. This claim stands under previously described assumptions on the nature of data in PLM, especially the simplicity of the pattern assumption.

In the fourth phase, overall system functionality was tested on two use cases, the manufacturing quality control domain of an industrial partner from the LinkedDesign project and the predictive maintenance domain of a diesel engine manufacturing company defined during a joint project. For quality control, the most relevant concern was modeling all stages of production that might carry a cause of defect in produced pieces. Using ontology reasoning, experience and routines of shop floor workers were modeled, to enable detection of "risky" conditions. Finally, through the data mining engine, the most probable factors of defect presence were identified. In the case of predictive maintenance, the focus was modeling the diesel engine maintenance process and its relevant factors. Ontology reasoning was exploited to adjust the maintenance period according to engines' middle of life characteristics. Finally, through data mining analysis, it was pointed out that machine failure can be predicted with certain accuracy.

In the fifth and final Phase, future improvements of this work are defined and possibilities for extensions are recognized. Entire content is revisited and discussed in the Conclusion with a walkthrough of the system functioning.

1.4 Contributions

There are three main contributions to this research all required for and leading to one single gain, a system that is automatic, self-initiated and autonomous in processing PLM domain data, resulting in underlying patterns detection, anomaly reporting and cause-effect relation detection. The three individual contributions required for such a system are:

USM method for PLM ontology design brings strategy for defining and analyzing a domain which needs to be modeled by ontology with consideration of further exploitation of ontology by reasoning engines and data mining engine. As the ontology represents a knowledge base, it needs to capture all relevant concepts, relations and dynamics of the domain. Be-

sides defining the procedural steps of this approach, in this research, the abstract, so called "upper ontology" was created to be used as a template graph for design of any domain specific ontology covered by PLM. This decreases the time requirements considerably and contributes to the wider acceptance of industrial ontologies in the future.

Ontological rules, as a method for shaping the functionality of ontology, and as a knowledge source are introduced. The manner in which ontology can act as a number of different services, such as, a decision support system, a quality control system, a knowledge base maintenance system and many more, was presented by using different use-cases. On the other hand, using ontology inference, concepts modeled in ontology will be interconnected with additional relations, compared to a manually predefined set. Having those relations automatically generated, underlines the benefits of using ontology as a source of tabular data to be examined. Using ontology inference, ontology becomes a more detailed model of reality than initially defined by human experts. Consequently, drawing data sets based on related concepts corresponds to gathering relevant data in real life.

Data mining engine (DME) is a novel design that brings data mining benefits to non-expert end-users. By relying on assumptions that data has been drawn from ontology, that processing speed is not vital, and that precision of pattern is not as relevant as detecting the pattern, it was possible to design a fully automated system. It implements learning algorithms, selected based on the PLM domain specificity and also, experts' data-miner's experience on tuning the parameters and hyper parameters of those algorithms. Results are delivered in human, and even more so, non-expert understandable form, yet still containing enough information to allow straightforward refinement of those results by experts. DME works as a "black-box", meaning that it doesn't require any understanding, involvement or input from an end user.

The information flow and structure of the contribution has been illustrated in Figure 1. Data sources, together with domain knowledge, are modeled into ontology in semantic module, using the USM method. The knowledge model is then enhanced by ontology inference and additional ontological functionalities are provided by rule reasoning. Tabular data sets are then exported and forwarded to data mining module. Within this engine, data sets are explored for patterns, dependences and out-of-ordinary data. Finally, relevant conclusions are delivered to an end-user.

Benefits for an end user are discussed for each of the contributions individually and finally for the overall system. It has been pointed out through use cases that many improvements are possible. Primarily, that the system has been designed as a conceptual generic solution which could perform better after specific task customization.

1.5 Thesis structure

In **Chapter 1**, the motivation for this thesis is introduced and the research questions and objectives are specified. Furthermore, a methodology for conducting this research work is presented and the contributions of the thesis are stated.

Part I contains literature review and knowledge gap definition for three different domains. As this Dissertation can be considered cross-disciplinary, it was important to define existing work and background knowledge for each of the domains independently and this was done in Chapters 2, 3 and 4.

In **Chapter 2**, the basic concepts and challenges of Product Life-Cycle Management (PLM) have been described. Starting from the definition of Product Life-cycle Management and its methods and issues up to the conclusion about open questions regarding data handling.

In **Chapter 3**, ontology comes into focus. Starting with a definition of knowledge and its representation, up to ontology as a key tool for knowledge structuring. This is followed by a list of relevant facts on the technical perspective of ontology implementation.

Chapter 4 is devoted to data mining as a discipline and the existing efforts to exploit populated ontology as a data set source. It finally lists the challenges of designing a fully automatic data set engine.

In **Part II** the contribution to this Dissertation is presented. The contribution refers to two different topics. The first topic covers the gap in design approach for domain specific ontology in a field of PLM. This contribution is presented as a semantic module definition and evaluation. The second topic covers the challenge of data mining procedure automation, using ontology and relaxed optimization goals. The contribution is presented through data mining module definition and evaluation.

Chapter 5 describes the first contribution, the semantic module, defined to exploit PLM issues and ontology benefits. User Story Mapping Method (USM) for ontology modeling is defined and elaborated on. Ontology reasoning is introduced and explained.

Chapter 6 contains tests of the semantic module, starting from the application of the USM method, to design domain specific ontologies, leading to the definition of ontological rules which enable diverse functionalities of ontology. The same procedure is conducted for three different use cases: Industrial standards; Life-cycle cost; and Knowledge management. At the same time, the second contribution to this Dissertation is presented through use cases.

Chapter 7 describes the third contribution to this research that is a data mining module. It contains details on every building block of the module architecture combined with a discussion on how ontology can be used to overcome known automation challenges.

The data mining module is thoroughly tested in **Chapter 8**. As a test bench, 6 data sets were used, well-known in the machine learning community as test-benches for new algorithms. Performance of the data mining module was compared to the best manually tuned solutions.

Finally in **Chapter 9, o**verall system testing is performed on two use cases, one from the Beginning of Life and one from the Middle of Life stages of a product. Results are discussed and potential improvements noted.

In **Chapter 10**, the conclusions of this Dissertation, as well as the future possibilities for extending the current work, are illustrated.

Part I : Literature review

Chapter 2 Product Life Cycle Management

2.1 PLM definition

Product design, manufacture, repair and recycle are concepts that have evolved since the early days of human history. Today, they are very complicated and knowledge intensive processes. Despite great advances in tools and techniques for product development and support, the central idea is still the same: determining a certain set of needs – actually coming from a customer – and developing a product that satisfies those needs. As quantity, complexity and variety of products increase, the one-person enterprise becomes inefficient and insufficient. During scale up, processes that formerly used to be performed by a single person are fragmented into several pieces each of which requires a simpler process, *i.e.*, simpler tools, less skill and less knowledge (Terzi, Bouras, Dutta, Garetti, & Kiritsis, n.d.). This happens for all processes of the product life-cycle, *i.e.*, design, manufacture, distribution, use, support, maintenance, repair, recycle and disposal. For the design and manufacturing phases, the reduction to simple unit operations gave rise to the mass production paradigm. Today, this paradigm is efficient for producing large quantities of products in automated and semi-automated plants.

Consequently, as concerns product design, today's knowledge-intensive product development environment requires a computational framework that enables the capture, representation and reuse of product and process knowledge. In the manufacturing phase, all this product information has to be shared along the production and distribution chain and synchronized with future updates. Moreover, product data is to be put at the disposal of the service chain during the use and support phases. During product use, input data on product behavior could be collected for design improvement. Recycling and dismissal activities may require and provide information on components, materials and other resources. Such sharing and managing of product data, information and knowledge forms the essence of the idea behind Product Lifecycle Management.

2.2 Closed loop concept

Closed loops ensure that valuable information is available to all lifecycle phases. The term "*lifecycle"* generally indicates the whole set of phases, which could be recognised as independent stages to be passed/followed/performed by a product, from 'its cradle to its grave'. Adopting an easy-to-use model, product lifecycle can be defined by three main phases (Kiritsis, Bufardi, & Xirouchakis, 2003) :

(1) Beginning of life (BOL) includes planning, design and manufacturing. Design is a multilevel phase since it comprises product, process and plant design. Generally, a design action implies a recursive application of multiple sub-actions: identifying requirements, defining reference concepts, doing a more and more detailed design, developing prototypes and performing tests. Manufacturing means production of the artefacts and relevant internal plant logistic. During this phase, the product is in the hands of the company within the boundaries of the enterprise (at least at its extended meaning). In the BOL phase, the product concept is generated and subsequently physically realised. Using many tools, techniques and methodologies, designers, planners and engineers develop the product design and the production process, plan the production facilities and manage manufacture of products with diverse suppliers (generally, through information sharing with an enterprise resource planning system).

(2) Middle-of-life (MOL) including distribution (external logistic), use and support (in terms of repair and maintenance). In this phase, the product is in the hands of the final customer, i.e. product user/consumer and/or some service providers, e.g. maintenance actors and logistic providers. In the MOL phase, products are distributed, used and supported (repaired and maintained) by customers and/or service providers. The product history related to distribution routes, usage conditions, failures and maintenance can be collected to create an up-to-date report about the status of products. Real-time preventive services, e.g. car control services, can be organised.

(3) End-of-life (EOL) where products are retired – actually recollected in the company's hands (reverse logistic) – in order to be recycled (disassembled, remanufactured, reused, etc.) or disposed. EOL is the phase where products are collected, disassembled, refurbished, recycled, reassembled, reused or disposed. It can be said that EOL starts from the time when the product no longer satisfies its users (the initial purchaser, a second hand owner, etc.). Information from EOL about 'valuable parts and materials' (what materials they contain, who manufactured them) and other knowledge that facilitates material reuse should be routed to recyclers and re-users, who can obtain accurate information about product status and product content.

The diagram from the same publication, mapping the information and control flow is given in Figure 3.

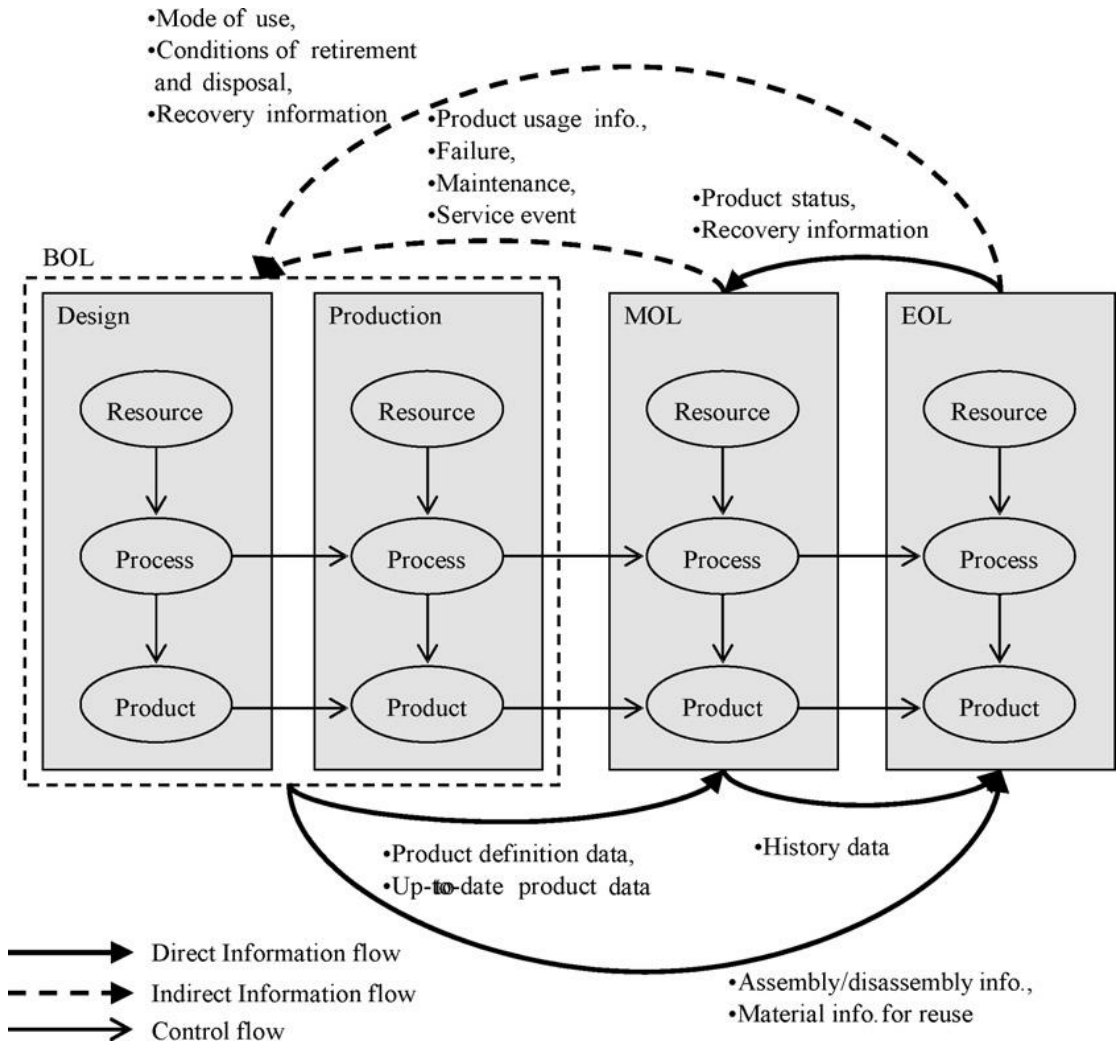


Figure 3 Information flow in PLM (Kiritsis et al., 2003)

2.3 PLM data

Product Data Management (PDM)– also known in recent years as Technical Data Management (TDM) or Engineering Data Management (EDM) – is basically a system for storing, archiving and managing product engineering data (*e.g.* drawings, design objects) and related workflows (Stark, 2005a). PDM is a useful tool for structuring and maintaining engineering data, required by other enterprise functions (e.g. manufacturing, planning, after sales) – and managing product configurations and variants. It provides an efficient tool for supporting releases and versions control and the engineering change process, coming from the factory.

Generally, a PDM system (Stark, 2005b) is composed of: (i) an information warehouse or vault where product data is stored in a structured way; (ii) an information management module, responsible for system administration, data accessibility, security and integrity, concurrent use of data, archiving and recovery; (iii) a workflow management module, to be used for defining workflows and registering workflow histories; (iv) a user interface, supporting user activities (queries, reporting, etc.) and (v) a series of system interfaces for programs such as CAD, CAE and ERP. PDM processes can be automated, not only for trivial activities, but also for more intellectual and value-adding phases, implementing Knowledge Based Engineering (KBE) techniques, derived by Knowledge Management practices (KM).

Data management tools like PDM are addressed as Collaborative Product Definition (CPD) systems. The set of authoring (CAx) and CPD tools are typically installed in R&D and engineering departments, for supporting the product development process. However, in its wider application, PDM systems could easily provide product data to diverse enterprise functions (*e.g.* manufacturing) and also outside the enterprise boundaries (providing product data to design and manufacturing suppliers).

Finally, data types, formats and structure managed within these tools are numerous in every stage of a product life-cycle. An attempt at giving an overview is shown in the following Figure 4.

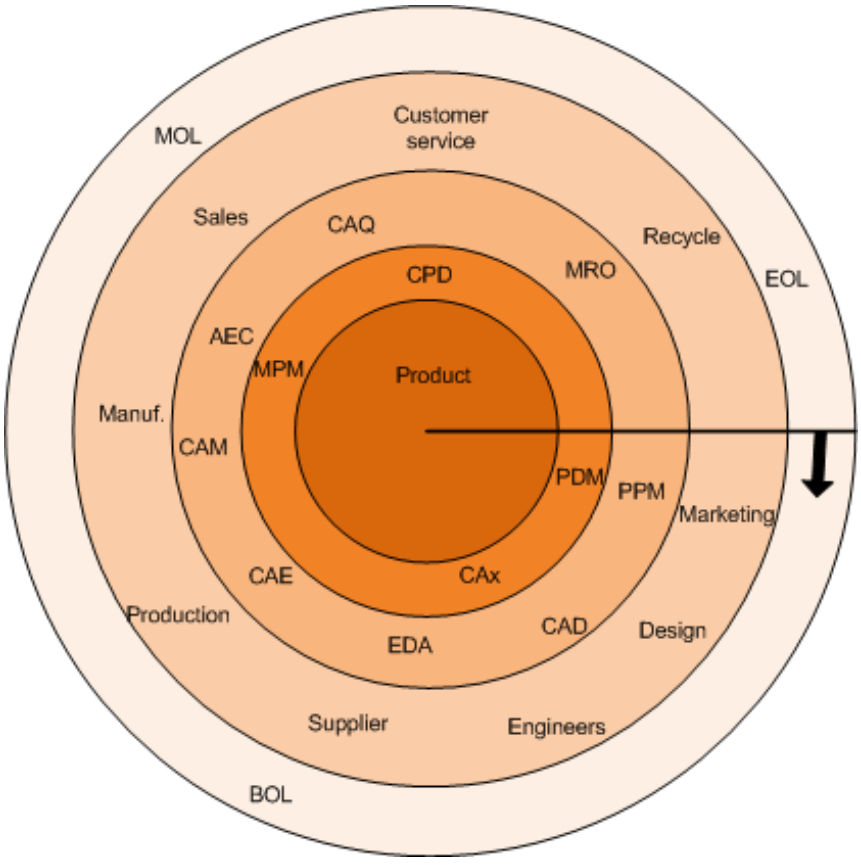


Figure 4 Overview of product data type

Starting with a marketing expert in the BOL phase, we can expect data to be generated in a Project Portfolio Management (PPM) tool. Further on, the designers will generate CAx files using CAD and CAID tools, engineers will use Electronic Design Automation (EDA) and Computer Aided Engineering (CAE). During the manufacturing phase we can expect data generated by tools such as Computer Aided Manufacturing (CAM) in Manufacturing Process Management (MPM). In MOL, sales will use Computer Aided Process Planning (CAPP) tools and Customer Service Computer Aided Quality (CAQ) control. Throughout all of these phases we can expect different data structures and metadata, as well as all groups of data formats (text, .pdf, images...).

All of these forms and formats are well known and well defined and can be represented in a form of interoperable semantic models.

2.4 Conclusion

The complexity of this system emphasizes the need for one unified model mapping all mentioned semantic models into one coherent structure. By organizing all the modules into one ontology, it ensures that updates are performed without causing inconsistency, information is visible throughout the domain and architecture of the knowledge base is designed so that it represents real life information flow. At the same time, it can be noticed that knowledge base is built so that information is available to everyone and with a wide variety of tools available to query and exploit data. In such a system, it is still left to an end user to hold the initiative and be aware of the contained knowledge and what he/she expects to find. This remains as gap in such a system having in mind that even an expert end user might be unaware of all relevant factors and correlation.

In this Dissertation, a data independent, autonomous data mining engine is proposed to address this gap. By relaying on assumptions about PLM domain uniqueness, it is able to provide novel knowledge about the domain to the end users.

Chapter 3 Ontology

The traditional goal of ontological inquiry in particular is to divide the world "at its joints" to discover those fundamental categories or kinds into which the world's objects naturally fall (Mayer et al., 1995). Without going into detail on how the definition of ontology is adapted for different fields like biology, geography and social sciences, we can turn to defining ontology within computer and information science interests. In theory, an ontology is a "formal, explicit specification of a shared conceptualization"(Gruber, 1993).An ontology renders shared vocabulary and taxonomy which models a domain with the definition of objects and/or concepts and their properties and relations. In other words, in computer science and information science, an ontology formally represents knowledge as a set of concepts within a domain, and the relationships between those concepts. It can be used to reason about the entities within that domain and may be used to describe the domain.

3.1 Knowledge definition

Knowledge is an elusive concept and therefore it is important to define it in context in order to understand it. The term is used in several ways in the literature. For example, Nonaka and Takuechi in (Nonaka & Takeuchi, 1995), two of the early researchers in this field, adopt a philosophical angle and define knowledge as "justified true belief". In this view, knowledge is an opinion, idea or theory that has been verified empirically and agreed upon by a community. According to Collinson in (Collinson, 2006) , knowledge at the most basic level is "that which is known". Klein et al in (Klein, 1998) liken knowledge with professional intellect where professional intellect in organizations centers on know-what, know-why, know-how and self-motivated creativity. Stewart in (Stewart & Ruckdeschel, 1998) also considers knowledge in terms of intellectual capital. On the other hand, Bohn in (Bohn, 1998) examines knowledge in terms of a company's processes. He believes that an organization's knowledge about its processes may range from total ignorance about how they work to very complex and formal mathematical models. According to Davenport et al in (T. H. Davenport & Prusak, 1998), knowledge is information combined with experience, context, interpretation and reflection. It is a high value form of information that is ready to apply to decisions and actions. Simply put, knowledge can be defined as the integration of ideas, experience,

intuition, assertions, skills and lessons learned that have the potential to create value for a business by informing decisions and improving performance (Wiig, 1997). In this view, knowledge is a key enabler to organizational success. However, in order for knowledge to be useful it must be available, accurate, effective and accessible.

3.2 Knowledge representation

Innovation is the application of knowledge to produce new knowledge (Drucker & Drucker, 1994). It requires systematic efforts and a high degree of organization. As we enter the knowledge society, ownership of knowledge and information as a source of competitive advantage is becoming increasingly important. In other words, organizations depend more on the development, use and distribution of knowledge-based competencies. This is particularly relevant in knowledge intensive processes such as product innovation. Consequently, research and development (R&D) organizations are paying more attention to the concept of managing their knowledge base and tools in order to increase competitive advantage, through effective decision making and increased innovation (Nonaka & Takeuchi, 1995)(T. Davenport, De Long, & Beers, 1998)(Sveiby, 1997). Knowledge is a key resource that must be managed if improvement efforts are to succeed and businesses are to remain competitive in a networked environment (Gunasekaran, 1999). In particular, the two major challenges that face organizations are: (a) ensuring that they have the knowledge to support their operations and (b) ensuring that they optimize the knowledge resources available to them. Managing knowledge is about creating an environment that fosters the continuous creation, aggregation, use and reuse of both organizational and personal knowledge in the pursuit of new business value. In short, the overriding purpose of enterprise knowledge management is to make knowledge accessible and reusable cross disciplinary and independent of time and location.

Application of knowledge engineering in product information management context required that the format used for representing the knowledge is understandable by both, humans and machines. For this reason, number of methods was developed, including relational diagrams and linked tables but lately, ontologies are shown to be preferable choice. In theory, ontology is a "formal, explicit specification of a shared conceptualization" (Gruber, 1993) . Ontology renders shared vocabulary and taxonomy which models a domain with the definition of objects and/or concepts and their properties and relations. In other words, in computer science and information science, an ontology formally represents knowledge as a set of concepts within a domain, and the relationships between those concepts. It can be used to reason about the entities within that domain and may be used to describe the domain. It is a common language between different actors and bridge for knowledge exchange. This schematic representation of knowledge makes it more understandable for humans, com-

pared to other semantic representations of objects and relations. Ontological tools require every concept and relation to be semantically defined and structured, which makes ontology machine-understandable. If populated, ontologies have shown to be very convenient for organizing and storing the data. This enables automatic reasoning and inference which means that beside the knowledge gathered in the time of modeling the ontology, additional relations will be automatically build up in time. In the perspective of selecting an ontology as knowledge representation method, capturing domain knowledge needs to lead to definition of the domain concepts.

Specification and conceptualization of ontologies lean on the identification of the relevant concepts of a particular domain, their type, and the constraints on their use. Existing methodologies such as Diligent (Casanovas, Casellas, Tempich, Vrandečić, & Benjamins, 2007), Methontology (Fernández-López, Gómez-Pérez, & Juristo, 1997), or On-To-Knowledge (Staab & Studer, 2004) lack detailed and clear guidelines for building the concepts. It is important to emphasize that, the process of concepts definition represents a key issue for knowledge gathering as it has to cover in an optimal way the whole domain. On the other hand, several knowledge resources may exist and their concepts reuse can be of a key importance.

The NeOn Methodology (Suárez-Figueroa, 2010) comes to deal with the aforementioned issues and provides some methodological guidelines for performing the ontology requirements specification activity, to obtain the requirements that the ontology should fulfill. Particularly, it consists of elaborating an ORSD (Ontology Requirements Specification Document) which aims to list, among others, the intended uses, the end-users and a set of questions describing the requirements that the ontology should fulfill. NeOn methodology responds to main requirements set up by characteristics of product information management domain. First of all it is scenario specific, meaning that ontology building is a bottom-up process which corresponds to scenario where overall perspective is not available. Further on, NeOn supports ontology network building, in a sense that the domain is distributed and defined by number of different actors. Finally, it supports collaborative building of ontology with NeOn Glossary of processes and activities for merging and re-use of ontological resources.

Nevertheless, this approach of listing intended uses and questions that the ontology should respond to may appear as a flat structure in the sense that it doesn’t lead to study. It doesn’t analyze the domain in terms of interactions that link the end-users and usages, before going in deep into the questions that the concepts should be able to answer.

3.3 Structure of ontology

Individuals (instances) are the basic components of ontology. The individuals in ontology may include concrete objects as well as abstract individuals such as numbers and words. Instances are matter in ontology and all other components exist to give structure, sense and meaning to instances. One of the general purposes of ontology is to provide a means of classifying individuals, even if those individuals are not explicitly part of that ontology.

Classes can be defined as abstract groups, sets, or collections of objects. Classes may classify individuals, other classes, or a combination of both. Classes are used to build a hierarchy of concepts from the most specific (as simple groups of individuals) to the most abstract and generic concepts, using superclass-subclass relations. It is particularly important for the PLM domain that this hierarchy is properly defined. The terms chosen for each level of concepts, will represent a shared vocabulary for all participants independent and thus need to cover every required granularity.

Relationships (also known as relations or property) between objects in an ontology specify how objects are related to other objects. Typically a relation is of a particular type (or class) that specifies in what sense the object is related to the other object within that ontology. Much of the power of ontologies comes from the ability to describe relations. Together, the set of relations describes the semantic of the domain. The set of used relation types (classes of relations) and their subsumption hierarchy describe the expressive power of the language in which the ontology is expressed.

Datatype properties are a specific type of binary relation where one object is a class and the second one is of the standard data types (numeric, string...). These are used to define attributes of a class which correspond to relevant aspects of real-life individuals gathered under a class. Datatype properties are actually carriers of the data for an ontology. When the ontology is populated with instances, in other words, when the raw data is stored in the ontology, actual values will be stored under associated datatype properties.

A component which raised many discussions about whether it should be adopted as part of the ontological structure or if it is a layer on top of ontology are ontological rules (Thomas Eiter, n.d.). No matter to which side of the discussion one is leaning towards, the rules remain as a mechanism which expands the functionality of the ontology by modeling logic and dynamics of the domain. More details about ontology rules will be discussed in the following chapters.

In this Dissertation, when drawing tabular data from ontology, instances take rows and datatype properties take columns. Thus, when drawing data sets from ontology, although a step from structured to unstructured data, it has the benefit of providing organized, concept centered data sets.

3.4 Ontology roles

The application of ontology is a complex domain to grasp as it covers every problem setup where structuring, exchange and re-usage of knowledge can be used. The first attempts to categorize application of ontology in information science were done only recently (EL Kadiri, 2015). We can define seven different abstract roles:

Role 1: Trusted Source of Knowledge

Ontology is a description of the concepts and relationships that exist, expressing a shared understanding of a domain that is agreed upon by a community. In addition to the representation of concepts, their properties and relationships, ontology incorporates the principles and axioms describing a reality in the form of general facts, often called rules, axioms or formulae. We consider that the main and basic role of ontology is the provision of a trusted and common source of knowledge, used and shared by a community.

Role 2: Data Base

Ontology in the 1st role as a trusted source of knowledge is considered as a reference framework. The resulting logic-based representations form a conceptual model that can help with storage, management and sharing of data. Thus ontology can play an additional role that is the role of a data base.

Role 3: Knowledge Base

Ontology rules and axioms defining the semantics of a domain are represented with logic languages, such as descriptive logic or first order logic. They are considered as *intentional knowledge* or also *explicit knowledge.* Being machine understandable, the ontology can simultaneously play an additional role of a knowledge base and support the deduction of implicit knowledge by processing logic-based rules using an inference engine.

Role 4: Bridge for Multiple Domains

Eliminating the need for repetition of entire design process for every application domain is worthy of consideration and can be possible by leveraging external resources. Through manual, semi-automatic or automatic ontology-based mechanisms such as mapping, align-

ment, specialization and merging, it is possible to adopt and extend existing ontological resources and meta-data initiatives being standards-based, bridging thus multiple domains from design, manufacturing, assembly, etc.

Role 5: Mediator for Interoperability

As a reference framework, ontology can serve as a basis for schema matching to support systems interoperability in close environments where systems, tools and data sources have no common recognition of data type and relationships. Besides ontologies are parts of the W3C[1] standards stack for the Semantic Web.

Role 6: Contextual Search Enabler

Ontology can play the role of an enabler of a contextual search engine answering complex and cross-domain questions without any specific knowledge about the data-source itself and through a common language for querying. Known Semantic Web applications intended to develop ontology searchers, are for instance Watson[2], OntoSearch[3] or Swoogle[4].

Role 7: Linked Data Enabler

The application of ontologies has grown with the advent of the Linked Data paradigm. Linked Data consists of the creation of data stores, called triple stores, using URIs for identifying resources and their relations. The application of the Linked Data principles appears to be a very promising approach in data integration.

3.5 Ontology implementation

The aim of using IT methods and tools for developing ontologies is to represent data in both a machine-understandable and a human understandable manner. Moreover, the use of the methods and tools may also be used in order to transform data into information and then into knowledge. In this chapter we present a brief description of the existing technologies, methods and tools for developing ontologies. The aim is to demonstrate the capabilities of

[1] World Wide Web Consortium
[2] http://watson.kmi.open.ac.uk/WatsonWUI/
[3] http://www.ontosearch.com/
[4] http://swoogle.umbc.edu/

the existing technology and the opportunities they would provide in PLM systems, when implemented.

3.5.1 Semantic Representation & Languages

3.5.1.1 Resource Description Framework (RDF)

The Resource Description Framework[5] (RDF) is a language for representing information about resources in the World Wide Web. It is particularly intended for representing metadata about Web resources, such as the title, author, and modification date of a Web page, copyright and licensing information about a Web document, or the availability schedule for some shared resource. However, by generalizing the concept of a "Web resource", RDF can also be used to represent information about things that can be *identified* on the Web, even when they cannot be directly *retrieved* on the Web. Examples include information about items available from on-line shopping facilities (e.g., information about specifications, prices, and availability), or the description of a Web user's preferences for information delivery.

RDF is intended for situations in which this information needs to be processed by applications, rather than being only displayed to people. RDF provides a common framework for expressing this information so it can be exchanged between applications without loss of meaning. Since it is a common framework, application designers can leverage the availability of common RDF parsers and processing tools. The ability to exchange information between different applications means that the information may be made available to applications other than those for which it was originally created.

RDF is based on the idea of identifying things using Web identifiers (called *Uniform Resource Identifiers*, or *URIs*), and describing resources in terms of simple properties and property values. This enables RDF to represent simple statements about resources as a *graph* of nodes and arcs representing the resources, and their properties and values. To make this discussion somewhat more concrete as soon as possible, the group of statements "there is a Person identified by http://www.w3.org/People/EM/contact#me, whose name is Eric Miller, whose email address is em@w3.org, and whose title is Dr." could be represented as the RDF graph in Figure 5.

It has to be noted though that like HTML, this RDF/XML is machine process-able and, using URIs, can link pieces of information across the Web. However, unlike conventional hypertext, RDF URIs can refer to any identifiable thing, including things that may not be directly

[5] http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

retrievable on the Web (such as the person Eric Miller). The result is that in addition to describing such things as Web pages, RDF can also describe cars, businesses, people, news events, etc. In addition, RDF properties themselves have URIs, to precisely identify the relationships that exist between the linked items.

Figure 5: An RDF Graph Describing Eric Miller[6]

3.5.1.1.1 RDF Vocabulary Description Language 1.0: RDF Schema (RDFS)

RDF provides a way to express simple statements about resources, using named properties and values. However, RDF user communities also need the ability to define the *vocabularies* (terms) they intend to use in those statements, specifically, to indicate that they are describing specific kinds or classes of resources, and will use specific properties in describing those resources. RDF itself provides no means for defining such application-specific classes and properties. Instead, such classes and properties are described as an RDF vocabulary, using extensions to RDF provided by the RDF Vocabulary Description Language 1.0: RDF Schema (RDFS)[7].

RDF Schema does not provide a vocabulary of application-specific classes like *exterms:Tent*, *ex2:Book*, or *ex3:Person*, and properties like *exterms:weightInKg*, *ex2:author* or *ex3:JobTitle*. Instead, it provides the facilities needed to *describe* such classes and properties, and to indicate which classes and properties are expected to be used together (for example, to say that the property *ex3:jobTitle* will be used in describing a *ex3:Person*). In other words, RDF Schema provides a *type system* for RDF. The RDF Schema type system is similar in some respects to the type systems of object-oriented programming languages such as Java. For example,

[6] http://commons.wikimedia.org/wiki/File:Rdf_graph_for_Eric_Miller.png
[7] http://www.w3.org/TR/2004/REC-rdf-schema-20040210/

RDF Schema allows resources to be defined as instances of one or more *classes*. In addition, it allows classes to be organized in a hierarchical fashion; for example a class *ex:Dog* might be defined as a subclass of *ex:Mamm*al which is a subclass of ex:Animal, meaning that any resource which is in class *ex:Dog* is also implicitly in class *ex:Animal* as well. However, RDF classes and properties are in some respects very different from programming language types. RDF class and property descriptions do not create a straightjacket into which information must be forced, but instead provide additional information about the RDF resources they describe. This information can be used in a variety of ways.

The RDF Schema facilities are themselves provided in the form of an RDF vocabulary; that is, as a specialized set of predefined RDF resources with their own special meanings. The resources in the RDF Schema vocabulary have URIrefs with the prefix http://www.w3.org/2000/01/rdf-schema# (conventionally associated with the QName prefix rdfs:). Vocabulary descriptions (schemas) written in the RDF Schema language are legal RDF graphs. Hence, RDF software that is not written to also process the additional RDF Schema vocabulary can still interpret a schema as a legal RDF graph consisting of various resources and properties, but will not "understand" the additional built-in meanings of the RDF Schema terms. To understand these additional meanings, RDF software must be written to process an extended language that includes not only the rdf: vocabulary, but also the rdfs: vocabulary, together with their built-in meanings.

3.5.1.2 OWL Web Ontology Language

OWL is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called an ontology. OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and thus OWL goes beyond these languages in its ability to represent machine interpretable content on the Web[8]. OWL is a revision of the DAML+OIL web ontology language incorporating lessons learned from the design and application of DAML+OIL.

OWL is directly linked with the vision of semantic web (Berners-Lee, Hendler, & Lassila, 2001) in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the web. As stated by W3C, the Semantic Web will build on XML's ability to define customized tagging schemes and RDF's flexible approach to data representation. In this sense, the first level above RDF required for

[8] http://www.w3.org/TR/owl-features/

the Semantic Web is an ontology language that can formally describe the meaning of terminology used in Web documents. A number of possible use cases for OWL is also provided by W3C[9].

OWL provides three increasingly expressive sublanguages designed for use by specific communities of implementers and users. Each of these sublanguages is an extension of its simpler predecessor, both in what can be legally expressed and in what can be validly concluded. The following set of relations hold. Their inverses do not.

- Every legal OWL Lite ontology is a legal OWL DL ontology.
- Every legal OWL DL ontology is a legal OWL Full ontology.
- Every valid OWL Lite conclusion is a valid OWL DL conclusion.
- Every valid OWL DL conclusion is a valid OWL Full conclusion.

Ontology developers adopting OWL should consider which sublanguage best suits their needs. The choice between OWL Lite and OWL DL depends on the extent to which users require the more-expressive constructs provided by OWL DL. The choice between OWL DL and OWL Full mainly depends on the extent to which users require the meta-modeling facilities of RDF Schema (e.g. defining classes of classes, or attaching properties to classes). When using OWL Full as compared to OWL DL, reasoning support is less predictable since complete OWL Full implementations do not currently exist.

OWL Full can be viewed as an extension of RDF, while OWL Lite and OWL DL can be viewed as extensions of a restricted view of RDF. Every OWL (Lite, DL, Full) document is an RDF document, and every RDF document is an OWL Full document, but only some RDF documents will be a legal OWL Lite or OWL DL document. Because of this, some care has to be taken when a user wants to migrate an RDF document to OWL. When the expressiveness of OWL DL or OWL Lite is deemed appropriate, some precautions have to be taken to ensure that the original RDF document complies with the additional constraints imposed by OWL DL and OWL Lite.

3.5.1.2.1 OWL Lite

Owl Lite [10] supports those users primarily needing a classification hierarchy and simple constraints. For example, while it supports cardinality constraints, it only permits cardinality values of 0 and 1. It should be simpler to provide tool support for OWL Lite than its more expressive relatives, and OWL Lite provides a quick migration path for thesauri and other taxonomies. OWL Lite also has a lower formal complexity than OWL DL.

[9] http://www.w3.org/TR/webont-req/
[10] http://www.w3.org/TR/owl-ref/#OWLLite

3.5.1.2.2 OWL DL

OWL DL[11] supports those users who want the maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). OWL DL includes all OWL Language constructs, but they can be used only under certain restrictions, e.g. while a class might be a subclass of many classes, a class cannot be an instance of another class. OWL DL is so named due to its correspondence with description logics, a field of research that has studied the logics that form the formal foundation of OWL.

3.5.1.2.3 OWL Full

OWL Full[12] is meant for users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. E.g. in OWL Full, a class can be treated simultaneously as a collection of individuals and as an individual in its own right. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary. It is unlikely that any reasoning software will be able to support complete reasoning for every feature of OWL Full.

3.5.1.3 Semantic Rule Languages

Rules are widely recognized to be a major part of the frontier of the Semantic Web, and critical to the early adoption and applications of knowledge-based techniques in e-business, especially enterprise integration and B2B e-commerce. This includes knowledge representation (KR) theory and algorithms; mark-up languages based on such KR; engines, translators, and other tools; relationships to standardization efforts; and, not least, applications. Interest and activity in the area of Rules for the Semantic Web has grown rapidly over the last years.

Known rule systems fall into three broad categories: first-order, logic-programming, and action rules. These paradigms share little in the way of syntax and semantics. Moreover, there are large differences between systems even within the same paradigm.

3.5.1.3.1 Rule Interchange Format (RIF)

The Rule Interchange Format[13] (RIF) is a W3C supported standard for exchanging rules among rule systems, in particular among Web Rule Engines. RIF is focused on exchange rather than trying to develop a single one-fits-all rule language because, in contrast to other Semantic Web standards, such as RDF and OWL, it is clear by the involved working groups that a single language would not satisfy the needs of many popular paradigms for using rules

[11] http://www.w3.org/TR/owl-ref/#OWLDL
[12] http://www.w3.org/TR/owl-ref/#OWLFull
[13] http://www.w3.org/TR/2010/NOTE-rif-overview-20100622/

in knowledge representation and business modeling. But even rule exchange alone is recognized as a daunting task.

Regarding RIF, The approach taken by the Working Group was to design a family of languages, called *dialects*, with rigorously specified syntax and semantics. The family of RIF dialects is intended to be *uniform* and *extensible*. RIF uniformity means that dialects are expected to share as much as possible of the existing syntactic and semantic apparatus. Extensibility here means that it should be possible for motivated experts to define a new RIF dialect as a syntactic extension to an existing RIF dialect, with new elements corresponding to desired additional functionality. These new RIF dialects would be non-standard when defined, but might eventually become standards.

Because of the emphasis on rigor, the word *format* in the name of RIF is somewhat of an understatement. RIF in fact provides more than just a format. However, the concept of format is essential to the way RIF is intended to be used. Ultimately, the medium of exchange between different rule systems is XML, a format for data exchange. Central to the idea behind rule exchange through RIF is that different systems will provide syntactic mappings from their native languages to RIF dialects and back. These mappings are required to be *semantics-preserving*, and thus rule sets can be communicated from one system to another provided that the systems can talk through a suitable dialect, which they both support.

The RIF Working Group has focused on two kinds of dialects: *logic-based dialects* and dialects for *rules with actions*. Generally, logic-based dialects include languages that employ some kind of logic, such as first-order logic (often restricted to Horn logic) or non-first-order logics underlying the various logic programming languages (e.g., logic programming under the well-founded or stable semantics). The rules-with-actions dialects include production rule systems, such as Jess[14], Drools[15] and JRules[16], as well as reactive (or event-condition-action) rules, such as Reaction[17], RuleML[18] and XChange[19]. Due to the limited resources of the RIF Working Group, it defined only two logic dialects, the Basic_Logic_Dialect[20] (RIF-BLD) and a subset, the RIF Core Dialect[21], shared with RIF-PRD[22]; the Production Rule Dialect (RIF-PRD) is the only rules-with-actions dialect defined by the group. Other dialects are expected to be defined by the various user communities.

---

[14] http://www.jessrules.com/
[15] http://www.jboss.org/drools
[16] http://www-01.ibm.com/software/integration/business-rule-management/jrules/#
[17] http://ruleml.org/reaction/
[18] http://ruleml.org/
[19] http://reactiveweb.org/xchange/
[20] http://www.w3.org/TR/2010/REC-rif-bld-20100622/
[21] http://www.w3.org/TR/2010/REC-rif-core-20100622/
[22] http://www.w3.org/TR/2010/REC-rif-prd-20100622/

Present and future RIF dialects are expected to share datatypes, built-in functions, and built-in predicates as defined by RIF Datatypes and Built-Ins (RIF-DTB). In particular, the current dialects RIF-BLD, RIF-Core, and RIF-PRD all share the foundations of RIF-DTB 1.0.

3.5.1.3.2 Rule Markup Language (RuleML)

RuleML[23] constitutes a family of Web rule languages which contains derivation (deduction) rule languages, which themselves have a webized Datalog language as their inner core. Datalog RuleML's atomic formulas can be (un)keyed and (un)ordered. Inheriting the Datalog features, Hornlog RuleML adds functional expressions as terms. In Hornlog with equality, such misinterpreted (constructor-like) functions are complemented by interpreted (equation-defined) functions. These are described by further orthogonal dimensions "single- vs. set-valued" and "first- vs. higher-order". Combined modal logics apply special relations as operators to atoms with a misinterpreted relation, complementing the usual interpreted ones (Boley, 2006).

RuleML is a markup language developed to express both forward (bottom-up) and backward (top-down) rules in XML for deduction, rewriting, and further inferential-transformational tasks. A number of markup languages that are defined as part of RuleML are the following:

(1) Mathematical Markup Language[24] (MathML)
(2) DARPA Agent Markup Language[25] (DAML)
(3) Predictive Model Markup Language[26] (PMML)
(4) Attribute Grammars in XML[27] (AG-markup)
(5) Extensible Stylesheet Language Transformations[28] (XSLT)

3.5.1.3.3 A Semantic Web Rule Language Combining OWL and RuleML (SWRL)

The Semantic Web Rule Language (SWRL) is based on a combination of the OWL DL and OWL Lite sublanguages of the OWL Web Ontology Language with the Unary/Binary Datalog RuleML sublanguages of the Rule Markup Language (RuleML). The proposal extends the set of OWL axioms to include Horn-like rules. It thus enables Horn-like rules to be combined with an OWL knowledge base. A high-level abstract syntax is provided that extends the OWL abstract syntax described in the OWL Semantics and Abstract Syntax document[29]. An extension of the OWL model-theoretic semantics is also given to provide a formal meaning for OWL ontologies including rules written in this abstract syntax.

---

[23] http://ruleml.org/
[24] http://www.w3.org/Math/
[25] http://www.daml.org/
[26] http://www.dmg.org/pmml-v3-0.html
[27] http://www.dfki.uni-kl.de/~boley/xmlag/attgramm/
[28] http://www.w3.org/TR/xslt
[29] http://www.w3.org/TR/2004/REC-owl-semantics-20040210/

The proposed rules are of the form of an implication between an antecedent (body) and consequent (head). The intended meaning can be read as: whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold.

Both the antecedent (body) and consequent (head) consist of zero or more atoms. An empty antecedent is treated as trivially true (i.e. satisfied by every interpretation), so the consequent must also be satisfied by every interpretation; an empty consequent is treated as trivially false (i.e., not satisfied by any interpretation), so the antecedent must also not be satisfied by any interpretation. Multiple atoms are treated as a conjunction. Note that rules with conjunctive consequents could easily be transformed (via the Lloyd-Topor transformations) (Lloyd, 1987) into multiple rules each with an atomic consequent.

Atoms in these rules can be of the form C(x), P(x,y), sameAs(x,y) or differentFrom(x,y), where C is an OWL description, P is an OWL property, and x,y are either variables, OWL individuals or OWL data values. It is easy to see that OWL DL becomes undecidable when extended in this way as rules can be used to simulate role value maps (Patel-Schneider, 1989). A XML syntax is also given for these rules based on RuleML and the OWL XML presentation syntax. Furthermore, an RDF concrete syntax based on the OWL RDF/XML exchange syntax is presented. The rule syntaxes are illustrated with several running examples. Finally, we give usage suggestions and cautions.

3.5.2 Ontology implementation environmments overview

Focus is given on the new generation of ontology engineering environments, particularly, on NeOn Toolkit, Protégé, and TopBraid Composer. They have extensible, component-based architectures, where new modules can easily be added to provide more functionality to the environment.

The NeOn Toolkit [30] is an ontology engineering environment that supports the complete life cycle of large-scale ontology networks. In order to support such a broad ontology modeling functionality, it has an open and modular architecture, which the NeOn Toolkit inherits from its underlying platform, Eclipse. Eclipse is a very rich development environment, which is widely adopted in the programming world and which perfectly fits to the modeling paradigm for ontologies. It provides developers with a framework to easily create, publish and integrate new features into the NeOn Toolkit. A substantial number of so-called plug-ins has

[30] http://neon-toolkit.org/

been developed within and outside the NeOn consortium and are available at NeOn Toolkit homepage. The NeOn Toolkit is available as an installable core version with the basic ontology functionality such as editing, browsing, ontology and project management. Currently, the following versions are available:

(1) The basic NeOn Toolkit provides the core functionality for handling OWL 2 ontologies.
(2) The NeOn Toolkit extended configuration includes advanced functionality for managing rule based models and ontology mapping facilities based on commercial extensions.

TopBraid Composer [31] is a modeling environment for developing Semantic Web ontologies and building semantic applications. It is fully compliant with W3C standards and offers support for developing, managing and testing configurations of knowledge models and their instance knowledge bases. It is implemented as an Eclipse plug-in.

TopBraid Composer incorporates a flexible and extensible framework with a published API for developing semantic client/server or browser-based solutions that can integrate disparate applications and data sources. TopBraid Composer is available in three different versions: Free Edition, Standard Edition and Maestro Edition.

Protégé [32] is an open platform for ontology modeling and knowledge acquisition. It is an open source, standalone application with an extensible architecture. The core of this environment is the ontology editor, and it holds a library of modules that can be plugged, called plug-ins, to add more functions to the environment.

The main Protégé functions are to: load and save OWL and RDF ontologies; edit and visualize classes, properties, and SWRL rules; define logical class characteristics as OWL expressions; execute reasoners such as description logic classifiers; and edit OWL individuals for Semantic Web markup.

Protégé is available in different versions, each including different plug-ins, whose main difference is the ontology language that they support:

(1) Protégé version 3 supports OWL 1.0, RDF(S) and Frames.
(2) Protégé version 4 supports OWL 2.0.

[31] http://www.topquadrant.com/products/TB_Composer.html
[32] http://protege.stanford.edu/

For the ontologies implemented during this research Protege was used, namely for the easy-to-use interface, strong support from community and large number of reasoners compatible.

3.6 Conclusion

It can be noted that ontology engineering remains a computer and IT discipline. As such, experts for ontology design and implementation are usually not familiar with details and specificity of the PLM domain, thus it has to be done through collaboration with domain experts. This process is usually iterative and very time consuming.

To address this gap, this Dissertation has focused on the first phase of concepts definition, meaning domain understanding from the perspective of involved users as it represents the main basis for knowledge definition and conceptualization. The following chapter discusses the proposed approach for dealing with concepts definition based on one of the agile methods, called User Story Mapping.

Chapter 4 Data processing

4.1 Data mining

Generally, data mining (sometimes called data, pattern or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information. Data mining as a discipline gives a number of tools for resolving this issue. All of the algorithms are designed to detect correlations, underlying patterns or functions that generate data. The purpose of data mining practicing can be summarized in two major directions. First is that once the patterns are detected, they can be used to predict future events, following the same processes that generated the existing data. This approach opens up the field of predictive analysis. The second major purpose is to condense information, meaning, instead of storing large amounts of unstructured data, patterns that generate data can be stored in the form of models which are almost always many times smaller in volume.

There is a huge amount of data available in the Information Industry. This data is of no use until converted into useful information. The extraction of information is not a simple process, it involves other processes such as Data Cleaning, Data Integration, Data Transformation, Data Mining, Pattern Evaluation and Data Presentation. Once all these processes have been completed, we are in a position to use this information in many applications.

There are a number of different listings for types of tasks addressed by data mining. In general, they can be summarized as in (Fayyad, Piatetsky-Shapiro, & Smyth, 1996):

(1) Outlier detection – the identification of unusual data records, that might be interesting or data errors that require further investigation;
(2) Association rule learning – searches for relationships between variables. For example a supermarket might gather data on customer purchasing habits, using association rule learning, the supermarket can determine which products are frequently bought together and use this information for marketing purposes;
(3) Clustering – is the task of discovering groups and structures in the data that are in some way or another "similar", without using known structures in the data;

(4) Classification – is the task of generalizing known structure to apply to new data. For example, an email program might attempt to classify an email as "legitimate" or as "spam";
(5) Regression – attempts to find a function which models the data with the least amount of errors;
(6) Summarization – providing a more compact representation of the data set, including visualization and report generation.

4.2 Data mining and ontology

For more than a decade now, the research community had been working on exploiting the combined advantages of ontology and machine learning procedures. The research has taken two main directions, the first one being toward the usage of machine learning algorithms to extract ontology models from unstructured data and the second one being toward the usage of machine learning algorithms to gain new knowledge from existing ontology. The first direction is out of scope of this work due to the PLM domain complexity and variety of unstructured data formats and sources. The second has been mostly the focus of researchers within the web semantics domain. The main target in this community seems to be a classification of instances membership to a given concept, that enables ontology population (Aberer et al., 2007), (d'Amato, Fanizzi, & Esposito, 2008), (Fanizzi, d'Amato, & Esposito, 2012) .This comes as no surprise as the main purpose of semantic web technologies is in the semantic enrichment of unstructured data. Still, solutions proposed in this work do not go beyond ontology population and they do not consider exploitation of class knowledge.

On the other hand, usage of data mining in the PLM domain is not a novel thing and there are numerous examples of such research. In (Köksal, Batmaz, & Testik, 2011), authors give a detailed literature overview on data mining application for quality improvement in the manufacturing industry. Different proposed solutions are compared for each step of the data mining procedure, and while many high quality results are obtained for the modeling phase, the conclusion is that most of them lack data handling and data preprocessing solutions. Exploiting product related data is an open issue for many product life-cycle stages and researchers have mostly remained focused on a specific stage, and product, individually. Product exploitation has been analyzed in (Liao, Chen, & Hsu, 2009) for the beverage market. Authors use ontology to structure the data that has been manually gathered and then apply clustering and association rules to profile the customers' behavior. The proposed solution is a very novel one, and efficient, but remains applicable to a very narrow domain. Use case of a genetic algorithm applied to product life-cycle data from few product generation is illustrated in (Lachmayer, Mozgova, Sauthoff, & Gottwald, 2014) for smart Gentellignet products. The product development process was analyzed based on collected data in (Do, Bae, &

Park, 2015), with a focus on gathering and transforming data, while the exploitation was left to a commercial system.

4.3 Automation challenges

A number of decision support systems, within the field of business intelligence, has been developed (Campos, Stengard, & Milenova, 2005). They typically include tools such as reporting and querying software, online analytical processing, data warehouse, decision engineering, business performance management, etc. Some of the most popular software products are Eclipse, Pentaho, MicroStrategy, LogiXML, Sybase IQ, etc. The major downside is that these tools are more oriented toward answering given questions, meaning assisting human analysts, without a significant degree of initiative in extracting underlying patterns. A reason for this might be found in a complexity of data mining procedures, or better said, in a high customization of these procedures, necessary for every specific problem. The research community has been working on developing a generic system for data mining but all proposed solutions remain domain specific or require a list of very constraining assumptions about the data (Padhy, Mishra, & Panigrahi, 2012)(Asghar & Iqbal, 2009). In literature, only one example of drawing knowledge from ontology for data mining purposes can be found, but this solution was developed strictly for the text document clustering problem (Yuan & Sun, 2004). To our knowledge, applying business intelligence for reasoning over domain described by ontology has not yet been accomplished. This comes as no surprise since the expansion of ontological presence within the business world happened relatively recently and this trend is still on its way up to its peak point.

The steps in the data mining process can be defined in many alternative ways, all of them equally correct. In this Dissertation, steps are defined as:

(1) Data understanding - in this step the data miner studies the data, the way it has been collected and it's usually done in collaboration with a domain expert. This step is basically impossible to automate completely so that it matches a human experts quality of result;
(2) Data pre-processing - this step consists of data sampling, outlier and noise detection and handling of missing data. This step can be automated to a certain extent but not completely;
(3) Modeling - this step represents the selection of algorithm and the tuning of algorithmic parameters. This step is the most easily automated and there is a number of tools that already perform certain tasks in this step automatically;
(4) Evaluation - evaluation represents the final choice of a model and it has to be based on evaluation criteria which can be highly dependent on the modeling purpose. This

step is very challenging even for human experts and for machines, remains unsolved;

(5) Visualization - when it comes to result visualization, although there is a large market of different visualization tools, the appropriate choice depends on the type of model, type of result and finally the expertise of the end user.

Each of these steps can be automatized to a certain degree, but not completely. In the following chapters this problem will be discussed by addressing the challenges of each step individually.

4.4 Conclusion

There have been a number of discussions on whether the data mining process can ever be deterministic or it will always remain a somewhat artistic discipline (Faloutsos & Megalooikonomou, 2007) (De Bie, 2011). In general, the conclusion leans towards the latter. One of the most widely used arguments is that automated data mining is like taking a photo with an automatic camera, the quality of the results is not always as good as what a professional can achieve (Berry & Linoff, 1999).

Whenever engineering is faced with NP-hard or even beyond NP-hard problems, the only approach is to introduce heuristics that will reduce the search space. In this case, searching for the optimal data mining procedure for each given data set is considered an extremely hard problem thus heuristics have to be employed. In this Dissertation, the problem of data mining automation is tackled by introducing assumptions about data, present in the PLM domain, as well as the structural functionality of ontology.

Part II: Research contribution

Chapter 5 Semantic module definition

The semantic module combines two contributions to this Dissertation. First, it addresses the gap between ontology and PLM domain experts, and formalizes the communication between them through the USM methodology. Based on this formalization, and template ontology designed in this research, ontology design time requirements are improved and the process is simplified. Second, it shows the added value of implementing ontology for structuring knowledge in the PLM domain by introducing various functionalities that ontology can adopt through an ontology reasoning mechanism.

The process of capturing knowledge generated in the implicit form, over a period of an organization's operations, requires methodical communication and an exchange of information between a number of different actors. Deep understanding and common vocabulary have to be established between knowledge management (KM) experts and domain experts, since domain experts can often fail to recognize some of their personal experiences as valuable knowledge components. Moreover, KM experts have to gain a deep understanding of the domain, to be able to recognize and formalize all the relevant relations and dependences within an organization's operations. The domain can be highly complex, containing a number of operations and functionalities, from many different departments of an organization, and KM experts have to be able to grasp the overall structure in one model or schema. This may lead to problems such as non-harmonized terminology, information loss or an unclear hierarchy model within an organization. Addressing every relevant actor within an organization individually allows KM experts to detect inconsistencies in terminology or cause-effect relations between different departments and generate a model that spans the entire domain and imposes mutual understanding. Several methodologies have been elaborated on to guide knowledge acquisition activities and thus avoid omitting essential knowledge (Skarka, 2007), however they have resulted in flat structures, in the sense that they don’t lead to the study and analyses of the domain primarily in terms of interactions and cause-effect relations. The User Story Mapping (USM) (Patton, 2008) method, from agile software development, appears to be a promising approach to addressing the previously stated challenges. It is a user centric method, which allows the designers of software to learn about

what future users expect, besides helping the users express their overall demands in a functional view which is near to them.

The study performed in this research results, first, in the application of the USM method for domain knowledge acquisition, and second, in the provision of a form of reference schema covering the functionalities of a manufacturing organization. The template is created by generalizing schemas from three different organizations, from very diverse domains, so as to be able create an abstract reusable schema. This template simplifies the application of USM in every future organization, as KM experts can use it as a guideline for communication with domain experts, making it easier to recognize relevant actors and key-functionalities of a given organization.

In the product life-cycle management, ontologies were mainly introduced for overcoming the problems related to interoperability and exchange of data and knowledge between different actors in a life-cycle of a product, as well as to enable the reuse of knowledge(Power, 2007) (McKenzie-Veal, 2010). Considerable amount of work in this field has been done by academic community, but also within research departments of leading industrial companies for their own use (Guo, Pan, & Heflin, 2005). One of the most important advantages of the ontologies is that they are machine understandable and they allow automatic reasoning and inference. The mechanism of these tools is almost always based on the rule engines which try to derive the answers or some new knowledge from the existing base set of rules and the knowledge base (Wu, Barash, & Bartolini, 2007). This new knowledge can also be dynamic, in a sense that it can be triggered by an internal or external event and using this property, it is possible to design a number of domain specific functionalities for the ontology.

For ontology design improvements, the main idea is the usage of an approach from the agile software development called the User Story Mapping (USM) for knowledge domain definition (A Milicic, Perdikakis, & Kadiri, 2012). When a new software product is being developed, one of the first steps of the process is to document the idea. This usually results in a description of key features that the developed product will have, optionally including a short abstract, called the “elevator pitch” that will be used to advertise the product and show its value to the customer. After shortly documenting the idea, the next step is to develop a concrete list of action items or tasks, also called backlog, that need to be implemented in order to transform the idea into a concrete product. Unfortunately, such backlogs, though arranged in a priority order, are usually flat structures. This is a common case for traditional project management methods, which come from the area of software development, such as the “Waterfall method” (Hass, 2007). They help the team members to understand what needs to be done next, but unfortunately do not explain why it needs to be done and what the whole system or product does. Such approach can be compared with having the puzzle pieces, but not knowing what the whole picture should look like, not knowing what the final goal is. Within the waterfall methodology, a flat backlog list is created, which is then priori-

tized. The advantage is that it is easy to maintain, but it is difficult to give an overview of all details and functionalities that the developed product will offer (Sliger, 2006) . The USM in contrast, is a user centric approach that organizes the backlog (the knowledge) along scenarios and users. It gives an overview, which helps to think about the product as a whole. Furthermore, it shows a synthesized view of how the different involved parties (users) interact with the product and what their expectations are on how to use this product (their knowledge about the product). The process involves the definition of users (or user groups) and their expectations on the functionalities that the product will have.

The process of defining a backlog for software development can be also transferred to processes in the domain of knowledge definition. Interviewing actors of the organization will give pieces of domain knowledge, without knowing what the whole picture is. USM is a method for creating a good backlog, where actors are directed on how to formulate the description of their activities and functions, reducing the risk of misunderstanding due to different terminologies. It is a method for structuring the backlog so that every requirement is precisely positioned in the structured system functionality. Last but not least, it is a visual aid for KM experts, using which they can have an overview of an entire domain. As such, it is a communication bridge not only between KM experts and actors, but between actors among each other.

5.1 User Story Mapping Method

A user story map is an approach that organizes the backlog along scenarios and users. It answers the question how a user uses the product. It is important to clarify that here, the product is considered to be a cross disciplinary knowledge base. Defining usage by all relevant users, thus defines the domain knowledge, the existing one, needed one and produced one. In this manner, a method for software functionality requirements is transformed into method for domain knowledge definition. A backlog consists of several structure blocks as shown in Figure 6.

(1) Usage dimension – It describes how a user would use the product. It shows the sequence of steps that a user would perform when using the product. It is very important that usage steps cover the whole scope of the product usage.
(2) User dimension – This dimension defines the types of users that will use the developed product. This dimension helps to identify different users and the aspects of the product that will be interesting for those users.
(3) Backbone – This section describes the activities that a user performs within a usage step. The backbone describes the activities that a user performs using the developed product. This section is called backbone as it often represents the essentials of the

product and suits as a guideline for the definition of the user stories, which are actually a refinement of the backbone.

(4) User stories as backlog items – This is the actual placeholder for the user stories. The user stories are ordered vertically under each activity and represent a refined version of an activity. It is recommended that user stories follow the pattern “As <user> I want to <feature> so that <value>”. After all user stories have been defined, they also need to be prioritized, taking into account the value that this user story brings, the technical risk of implementing it and the effort that will be needed.

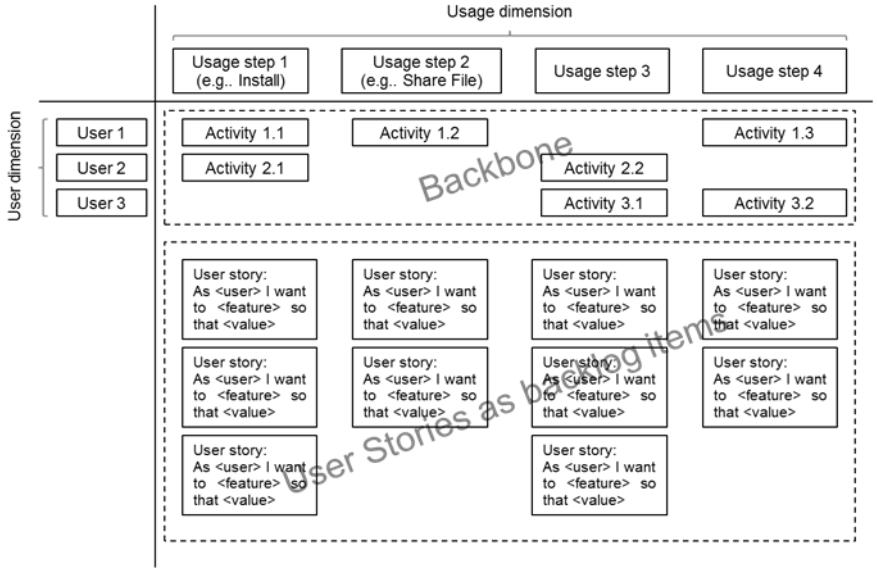


Figure 6. User story map (A Milicic et al., 2012)

Although the USM method was originally proposed and used in the area of agile software development, it is a generic method for structuring and sharing information about a product. It visualizes in a simple way several aspects of a product looking from the user perspective. In the context of collaborative product development, it can be used both as means of communication and as means of documenting knowledge, further to be exploited for ontology design purpose.

5.2 USM method for modeling ontology

If we consider the USM to be the first step of gathering information regarding the domain of interest, a following step can be defined as translating functional needs into list of concepts. As explained, if we select ontology as a knowledge representation tool, modeling requires that all relevant objects and factors are defined as concepts. After the application of the USM we have a detailed and structured information about knowledge present in the domain, how it is exploited and exchanged between actors. Creating concepts requires recog-

nizing leading objects and factors that will be translated directly into concepts. For example, in a manufacturing company, the term "machine" will be present in number of user stories, so it is clear that ontology will contain the concept "Machine". This concept will model all the knowledge about one machine, as well as its usage and functionalities.

In addition to user activities and experiences, valuable sources of knowledge are industrial standards and experiences from previous projects and organizations. Same procedure of recognizing key aspects for purpose of concepts definition can be applied here.

Finally, all these concepts have to be organized into network covering the entire domain. A key for structuring it can also be found in USM. Relations between concepts can be recognized as "cause-effect" dependences between concepts, as information flow connections, as co-operation between concepts, as "parent-child" set-up etc. These relations will tie domain concepts according to functionality of the organization, thus modeling knowledge about dynamics present. Based on this, the process of building a complete and structured knowledge base is illustrated in Figure 7:

Step 1: apply the USM method

Step 2: gather other sources of information (standards, past experience, etc.)

Step 3: create a unique list of concepts that covers entire domain

Step 4: define relations and dependencies among these concepts

Step 5: create a dynamic knowledge base covering the domain, expressed in some of the standard formats like relational data base, ontology, semantic model, etc.

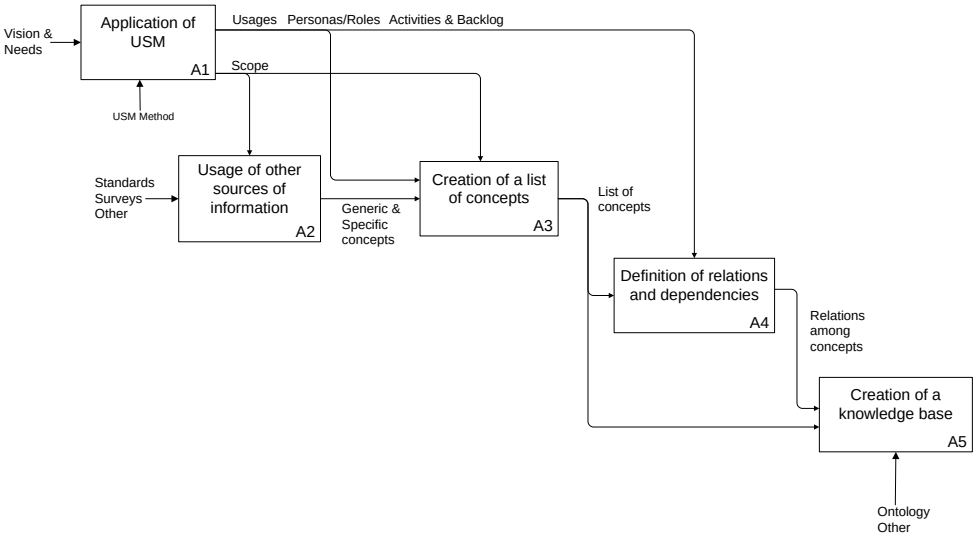


Figure 7. The proposed process for creation of knowledge base (A Milicic et al., 2012)

In this scenario, USM is vital part as it will create a base view of the domain in question. Switching from USM to list of concepts is relatively straight-forward step. Functionalities required by user stories are described in form of sets of functional modules and each module is translated into concept of the domain. Next, the list is extended with additional concepts coming from other sources of information like industrial standards or similar projects. Finally, concepts are described using relations and expressed in some of the usual knowledge base formats.

In the following chapters it will be shown how starting from USM application to various PLM specific domains, generalization was performed that lead to definition of upper ontology for PLM or the ontology template.

5.3 Ontology rules for modeling dynamics of the domain

Using the USM method to grasp knowledge of the domain will result in a graph of concepts, relations and properties, however not every knowledge can be expressed through the graph. Domain knowledge often contains dynamics between concepts, processes, which are triggered only when a certain set of conditions is met.

Ontology rules, as a component of ontological structure, enables modeling the dynamics of the domain, and functional and interdependent relations between concepts and their properties. In general, ontologies concentrate on classification methods, putting an emphasis on defining 'classes', and 'subclasses', on how individual resources can be associated to such classes, and characterizing the relationships among classes and their instances. Rules, on the other hand, concentrate on defining a general mechanism on discovering and generating new relationships based on existing ones, much like logic programs. In the ontology design phase, an initial set of rules is manually defined so that it implements logic, conditional processes and inherence structure. After that, it is left to inference engines (or reasoners) to automatically generate every valid conclusion. There are several approaches used by inference engines:

(1) Description logic based inference engine: They are used to perform basic reasoning tasks like consistency checking and subsumption concepts. It has the advantage of using decidability;
(2) First order logic (FOL): Reasoner takes an OWL file as input and first translated into FOL. Then inference is processed by using any one of the existing automated theorem prover;

(3) Combination of FOL and general Logic based inference engine: In this approach a fragment of FOL and general logic is used to design an OWL inference engine. Horn logic is most widely used due to its simplicity and availability.

Without going into details, we give an overview of relevant characteristics for most popular reasoners in Table 1.

	Pellet	**RACER**	**FACT++**	**Snorocket**	**HermiT**	**CEL**	**TrOWL**
Methodology	Tableau based	Tableau based	Tableau based	Completion rules	Hypertableau based	Completion rules	Completion rules
Soundness	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Completeness	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Rule Support	Yes (SWRL)	Yes (SWRL)	No	No	Yes (SWRL)	No	No
Platforms	all	all	all	all	all	Linux	all
Protégé Support	Yes	Yes	Yes	Yes	Yes	Yes	Yes
NeOn Support	Yes	No	No	No	Yes	No	No
Impl. Language	Java	LISP	C++	Java	Java	LISP	Java
Availability	Open source	Commercial	Open Source	Commercial	Open source	Open source	Commercial

Table 1 Overview of most popular reasoners (Abburu, 2012)

5.4 Ontology reasoning

For this project, we have chosen to use the Pellet reasoning engine. As with most of the engines, Pellet works on a knowledge base composed out of TBox (a semantic layer where ontology concepts are defined) and ABox (where instances are defined). Reasoning services applied to this knowledge base are defined as (Dalwadi, Nagar, & Makwana, 2012):

(1) Consistency Checking: consistency checks whether ABox instances are consistent with TBox concepts and ensures that ABox meets all of the restrictions;

(2) Satisfiability Checking: it checks the OWL concept Ccan have instances according to current ontology. That is ABox instances should be satisfied by available OWL concepts of TBox;

(3) Subsumption Checking: it checks whether a class D subsumes another class C. That is instances or properties of class C is also part of class D;

(4) Query Processing: OWL inference engines need powerful language to support queries so that users and software agents can query on knowledge base to retrieve useful data or facts;

(5) Reasoning with Rules: because the rules are capable of expressing OWL classes, properties and instances, an OWL inference engine needs to provide interface to process rules that are represented with OWL classes, properties and instance data.

5.5 Conclusion

Identification and representation of knowledge in a product related domain is still a great challenge. Here we have proposed a USM method for knowledge extraction as well as formal guidelines for applying this method. The USM method gives us a tool which can directly translate raw data into a list of relevant concepts that covers an entire functional profile of the software in question and thus it gives us a detailed image of the domain this software operates in. It is simple and straight-forward and it enables end users to express their descriptions of the domain in a common everyday language, rather than using technical terms, which is more likely to lead to the gathering of more detailed information. In addition to information formalization, it has been shown to be an excellent tool for generalization of "end user" requests and a vital step toward the creation of a knowledge base. Further on, the domain dynamics are represented by using ontology relations and reasoning engines, thus giving the full semantic model of a manufacturing domain. Through generalization, a reusable reference schema is created as an abstract ontology, which can be easily extended and implemented for the PLM domain of specific use cases. An entire process can be seen as a clear sequence of steps, forming a methodology for semantic modeling of the domain knowledge and data sources.

Chapter 6 Semantic module implementation

6.1 Design of upper PLM ontology using USM method

USM method gives an approach on “end user” request specification. In a real-life scenario, when creating a new product, there is always a dilemma between creating a generic product which can be used by everybody, but doesn’t really cover anybody’s needs completely or creating a strictly end-user dictated custom product. The USM method gives a solution for such problems since it allows a controlled generalization of user requests. This method for request specifications is developed for the scenario where some of the "end users" are already known and the product is developed according to their specifications, but since the controlled generalization of these requirements is done, it is possible and quite straight forward for other future clients to use the product. This is bottom-up approach that resulted in design of generic PLM ontology that can be used as template or a guideline schema for design of ontology for any domain ontology.

For this research, a number of different organization was selected, so that their activities and focus are as diverse as possible within PLM domain. Lists of concepts have been drawn from USM for these organization and the produced ontologies can be used as knowledge base for any software platform. The added value is created when these ontologies are merged by mapping of mutual concepts. This creates a set of generalized knowledge elements which are present in most of the manufacturing companies and thus, can be reused. Having this set of concepts shortens the time needed for knowledge acquisition for every future organization, since it can be used as guideline for what elements of knowledge are expected to be found. Additional analysis has to be done only for activities and functionalities, which are highly specific for a given organization. One of the questions that necessarily emerges is to what degree should some product be generic. The more generic it is, the wider is the spectrum of “end users” that will be able to use it. On the other hand, generic models

require more work on implementing them for specific companies. One of the approaches is to gather USMs from all interested users and then merge them into one single USM by generalizing them only to such extent, that the final model is simple enough. For example, when developing a software for different kinds of manufacturing companies, user roles such as “Part designer” and “Designer” coming from two different companies should be merged into one simple role “Design engineer” and both users should be able to identify the connection. In the context of this research, an example is shown in Figure 8.

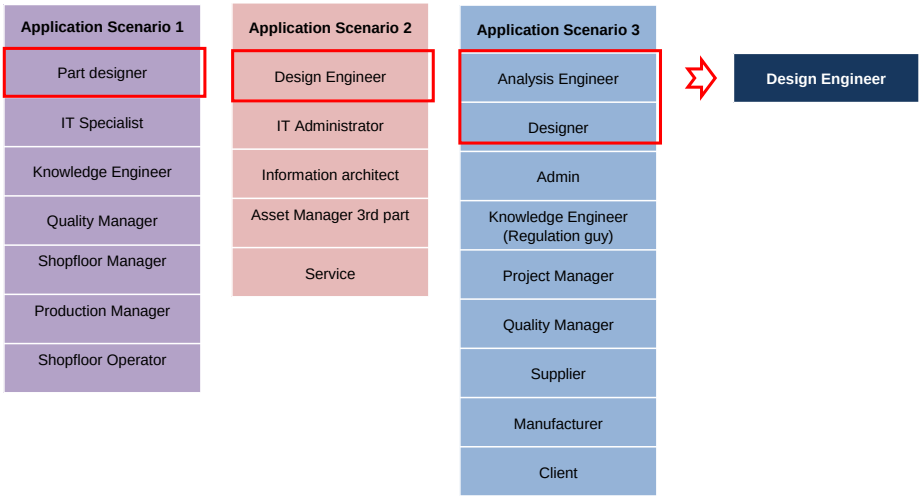


Figure 8. User roles merging (A Milicic et al., 2012)

In some other cases generalizing is not that simple. For example, if only one of selected companies need to have the role “Supplier”, it has to be carefully considered should such role be included in the generic model, or should it be taken under “Manufacturing personnel” which everybody requires. The later choice comes as better solution, since USM method gives us the opportunity to note the work of “Suppliers” through a user story of more generic role. These examples lead us to a conclusion that during this merging process, the most important thing is to have constant communication with all “end-users”. As long as they are able to recognize all their user stories in these merged, more generic roles, the model is not too generic.

Same guidelines stand for the creation of an optimal set of generalized functions, though in this case, the situation gets a bit more complicated as a consequence of difference in the nature of manufacturing that different clients implement. For example, function "Install" can be required by almost all clients but since there is no standardized terminology, "Install" can be used to describe installation of a knowledge base platform, overlooking the manufacturing or installation of a sensors overlooking the physical machinery. Function "Report" of a project manager is another one that can be expected from all clients but depending on a

case, it can be a request to be able to report feedback about product to a manufacturing department for adjusting the protocols or it can be a request to be able to report final statistics to a executives department for a future planning. This would be a problem of using the same word for different actions but also the problem of using different names for one action has to be considered. For example simple data processing can be described as "Analytics" or "Knowledge extraction" depending on a client. The only way to deal with these issues is again to have a continuous communication with all the clients and to make sure that terminology is well defined and understood.

General guidelines on how this process should be conducted are given in Figure 9. Still it is important to remember that this is a manual procedure and that human perception and creativity are crucial for success.

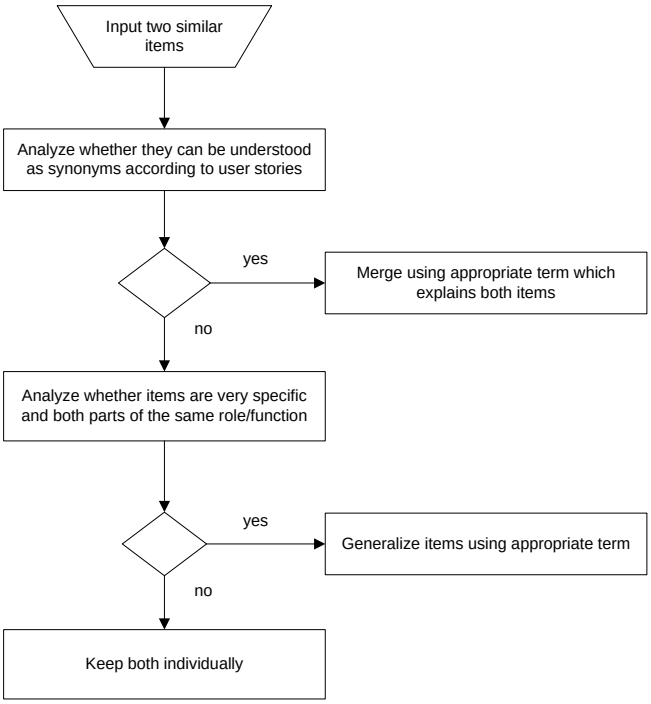


Figure 9 Algorithm for USM merging (A Milicic et al., 2012)

Another important issue to consider is that all actors have to fully understand what they are being asked for, by this USM method. In other words, they have to be able to describe their demands fully and in a detail manner and to be encouraged to consider all their employees and their activities. Close communication is again answer for this matter, since it will also lead to creating a more synchronized vocabulary and better understanding of terms used to describe different activities. Finally, the best results should be achieved through iterative

procedure of merging which is finished when final model is simple enough and it covers all demands for details of different users.

Once the generalized USM map is created, the procedure toward creating a list of concepts (referring to Step 3 of the proposed approach) is quite straight forward. Every scenario and process have to be covered with abstract models of its inputs. Every tool, machine, and raw material is for example modeled as sub-concept of generic concept Resource.

In the following Table 2 we give the resulting list of generated concepts as the first contribution of this dissertation.

No.	High Level Concepts	Description
1	Task	Groups scheduled actions
2	Process	Groups all processes
3	Product	Groups all products
4	Module	Groups modules within a product structure
5	Part	Groups parts within a product module
6	Resource	Groups all the required resources for product manufacturing and design
7	Actor	Groups all the persons involved in product development
8	LCP	Groups life-cycle phases of the product
9	Event	Groups a list of events
10	Factor	Groups relevant issues related to product which need to be examined
11	Indicator	Groups metrics applied to evaluate a factor

Table 2 High level concepts (A Milicic et al., 2012)

Concepts are defined to be generic enough so that they can cover the domain of any manufacturing organization, and yet, intuitive enough so that every actor and scenario from USM can be easily recognized in the network of concepts.

Further on, following "step 4" explained in previous chapter, dependencies and connections of the domain are modeled into relations between concepts. Full network of generic ontology and examples of relations defined are illustrated in Figure 10 and Table 3. Ontology implementation using Protege software is given in Figure 11. Every actor's activity, autono-

mous process and their details can be expressed as a combination of subject (concepts), connected by verb (relation) to an object (subject).

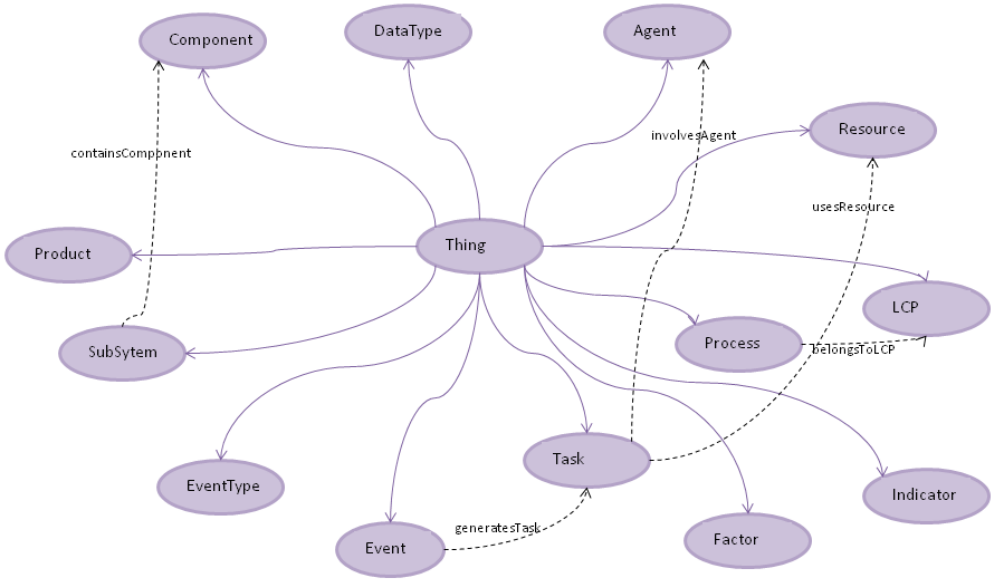


Figure 10 The upper PLM Ontology

No.	*Relation*	*Domain*	*Range*
1	*generatesTask*	Event	Task
2	*involvesAgent*	Task	Agent
3	*usesResource*	Task	Resource
4	*belongsToLCP*	Process	LCP
5	*containsComponent*	SubSystem	Component

Table 3 An excerpt of relations between concepts (A Milicic et al., 2012)

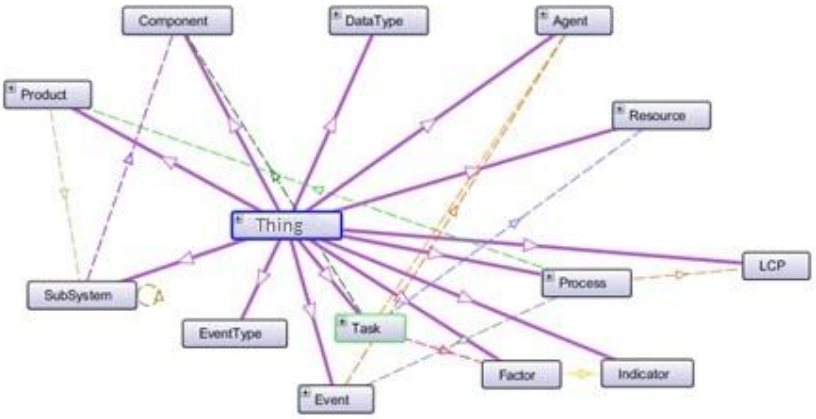


Figure 11 The upper PLM Ontology in Protege (Kiritsis, 2013)

For a complete domain spanning, this generic ontology needs to be extended with additional concepts covering knowledge which is highly specific for a given organization. Still, having this template as a starting point guaranties shortened ontology design time, as well as guidelines on how to structure the domain knowledge. At this point, it should be clarified that in this project ontologies are used as a tool for realizing the results of the proposed method. Ontologies are widely used for knowledge capturing and reuse and thus, here introduced to help us digitalize the final extract of the User Story Mapping method. The proposed model can be used to enable interoperable knowledge systems. In (Matsokis & Kiritsis, 2010) , in the context of Product Lifecycle Management (PLM), it is showcased how such a system simplifies knowledge intensive operations leveraging information coming from diverse sources (ERP, PLM, CAM/CAD, sensors, etc.) in the manufacturing domain.

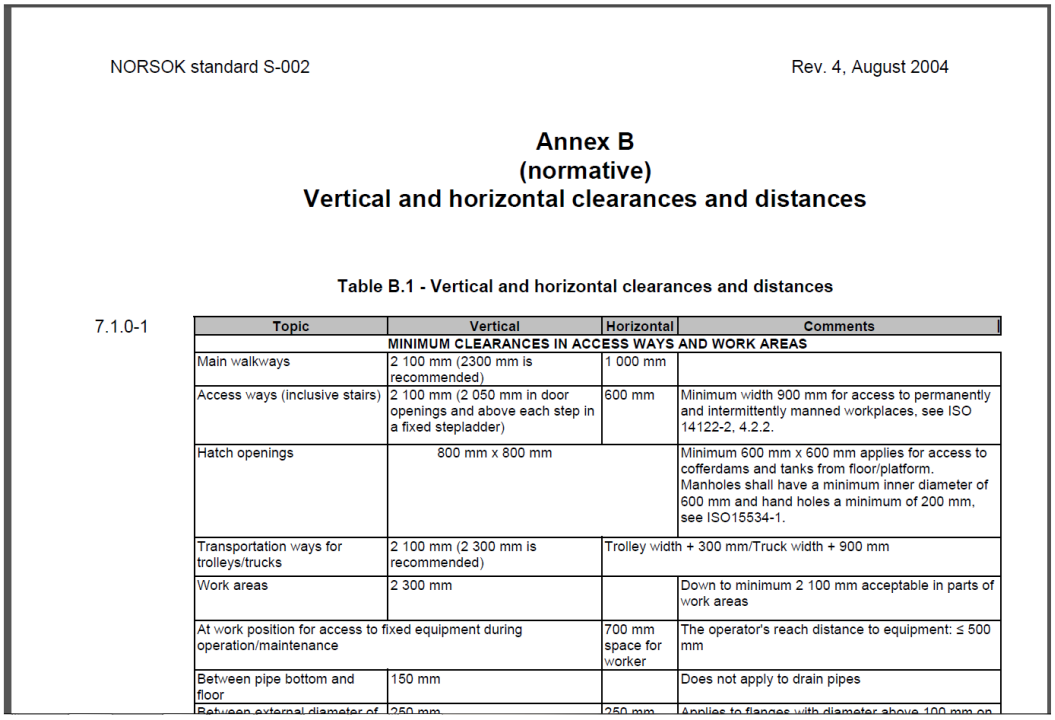
6.2 Design of domain specific ontologies using upper ontology

In this chapter, we tested the spanning of upper PLM ontology by applying it to three very different specific domains. All three use cases came from functional requirements of industrial partners in the LinkeDesign project. The process starts by analyzing a list of upper ontology concepts in order to select those that are relevant for the domain in question. Selected concepts are then elaborated on to fit better with the characteristics of the domain. Finally, additional concepts are added, so that the entire domain is spanned in details.

At the same time, the second contribution to this Dissertation is presented. It shows how a knowledge of dynamics in the domain can be modeled by ontology rules, so that it creates added values for ontology functionality. As it will be illustrated, ontology can act as an alert system, design consistency control, knowledge maintenance system and many more.

6.2.1 Use case on industrial standards

All major industrial projects rely on following industrial standards and directives published by governments or local institutions. These documents address a wide variety of issues such as environmental regulations, workers conditions regulations, safety issues and many more. Companies are obliged to consider and follow appropriate standards depending on a projects location, specific activities and main risks involved. The number of industrial standards is high and growing, and updated often. With today's fast developing competitor's market and short board-to-market project periods, it becomes obvious why addressing unstructured, large volumes of industrial standards is a bottle neck for the design stage of a project. The example used in this research is shown in Figure 12.



NORSOK standard S-002 Rev. 4, August 2004

Annex B
(normative)
Vertical and horizontal clearances and distances

Table B.1 - Vertical and horizontal clearances and distances

7.1.0-1

Topic	Vertical	Horizontal	Comments
MINIMUM CLEARANCES IN ACCESS WAYS AND WORK AREAS			
Main walkways	2 100 mm (2300 mm is recommended)	1 000 mm	
Access ways (inclusive stairs)	2 100 mm (2 050 mm in door openings and above each step in a fixed stepladder)	600 mm	Minimum width 900 mm for access to permanently and intermittently manned workplaces, see ISO 14122-2, 4.2.2.
Hatch openings	800 mm x 800 mm		Minimum 600 mm x 600 mm applies for access to cofferdams and tanks from floor/platform. Manholes shall have a minimum inner diameter of 600 mm and hand holes a minimum of 200 mm, see ISO15534-1.
Transportation ways for trolleys/trucks	2 100 mm (2 300 mm is recommended)	Trolley width + 300 mm/Truck width + 900 mm	
Work areas	2 300 mm		Down to minimum 2 100 mm acceptable in parts of work areas
At work position for access to fixed equipment during operation/maintenance		700 mm space for worker	The operator's reach distance to equipment: ≤ 500 mm
Between pipe bottom and floor	150 mm		Does not apply to drain pipes
Between external diameter of	250 mm	250 mm	Applies to flanges with diameter above 100 mm on

Figure 12 Extract from an industrial standard

6.2.1.1 Functional requirements for the use case

Considering this problem setting, ontology emerges as one of the solutions for creating a fast and automatic system for enforcing the application of industrial standards regulations.

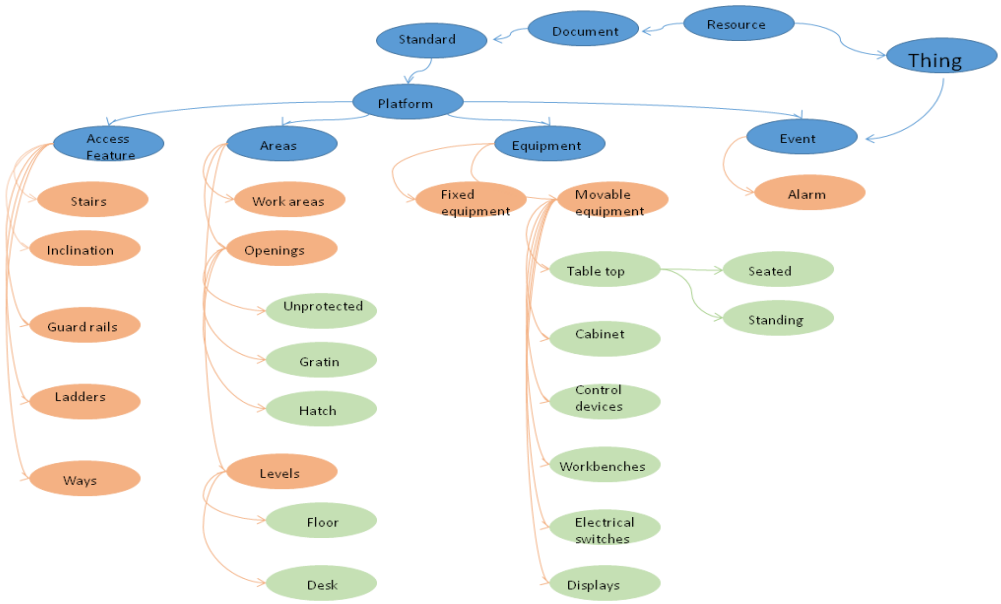


Figure 13 Norsok industrial standard ontology

Ontology designed for example of Norsok[33] industrial standard is presented in Figure 13 where the concepts present in the upper ontology are denoted with blue color. Properties of concepts are not presented since all the concepts have the same two attributes as it will be explained later.

The NORSOK standards have been developed by the Norwegian petroleum industry to ensure adequate safety, value adding and cost effectiveness for petroleum industry developments and operations. Furthermore, NORSOK standards, are as far as possible, intended to replace oil company specifications and serve as references to authorities regulations.
Norsok provides detailed directives for functional and design solutions for off-shore oil platforms, and, as an example of semantic modeling, directives for the design of Topside part were modeled in ontology.

[33] https://www.standard.no/en/sectors/energi-og-klima/petroleum/norsok-standards/#.VOytrfmvCUk

6.2.1.2 Ontology rules for the use case

Each of the concepts presented in Figure 13 have two attributes that is "Standardized" and "AsDesigned". For this use case, all rules have the same form that is, for each concept, values of parameters according to standards, and according to design solutions, are compared. In case of a violation, a warning message is sent out to the designer. Implemented ontology will be receiving "AsDesigned" values straight from the CAD system where the Topside is being designed. That creates an automated system for design control. The true value of such a solution, compared to the usual software solutions, is that ontology gives an overview of standards content in a structured manner, which allows for the efficient update or customization of the content. Implementation of all rules is given in Appendix A.

6.2.2 Use case on design recommendations

The second use case company focus involves mostly two different phases of the product lifecycle: the proposal (or concept) phase and the operational phase. The company's products are highly customized, full manufacturing lines and company provides, planning, manufacturing, delivery, installation and maintenance of these production lines. Having such products places a high importance on the planning and proposal phase since the company can only give approximate costs of their products during initial negotiations with a customer. Based on a customer's request, the product has to be outlined on the spot, and estimated costs have to be drawn from previous similar projects. The schema of information flow is given in Figure 14.

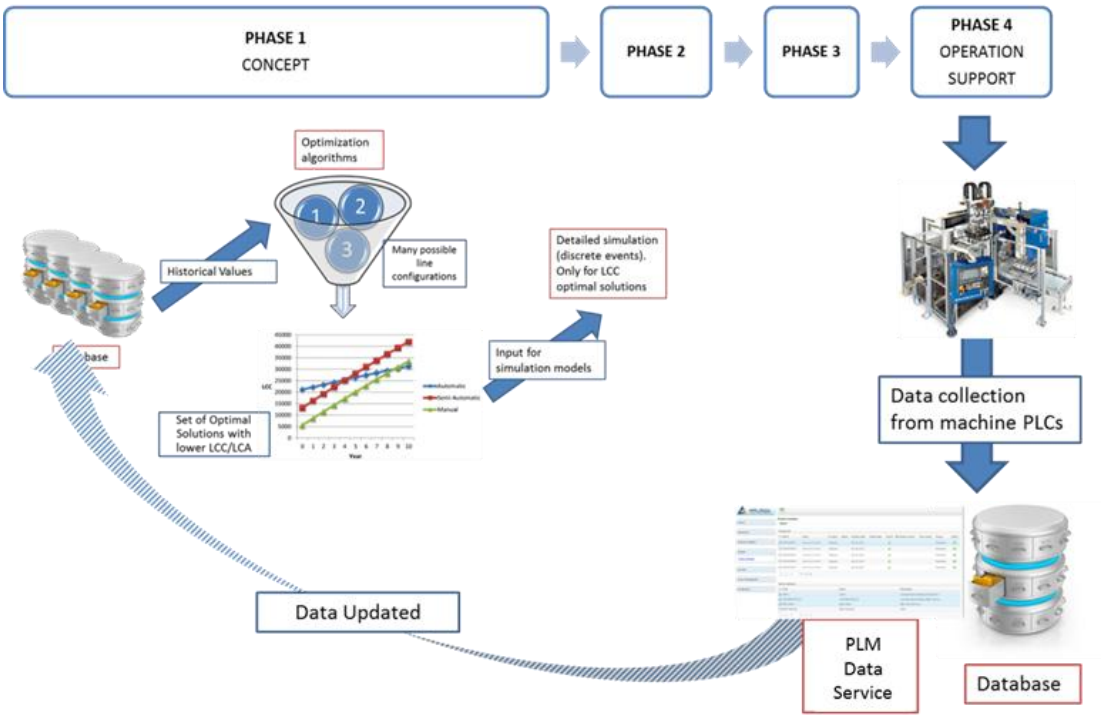


Figure 14 Use case company information flow [34]

6.2.2.1 Functional requirements for the use case

Ontology designed to cover such domain is given in Figure 15. The requirement of a company is to have an automated system to calculate the life-cycle cost (LCC) of a product (in this case, an engine assembly line) and to optimize the line configuration according to the cost and environmental impact. This is done in the proposal phase in such a manner so as to provide more information to the customer with an offer. However, data used to calculate the LCC come from databases that can be refined by collecting information from the shop-floor where the equipment is currently working. This connection between the operational phase and the concept phase, of the equipment, is reached through an automated data collection system able to record all the relevant information coming from the machine's control system (production data, failure data, maintenance activities, energy consumption, etc.).

[34] http://www.linkeddesign.eu/results/ (restricted document, available upon request)

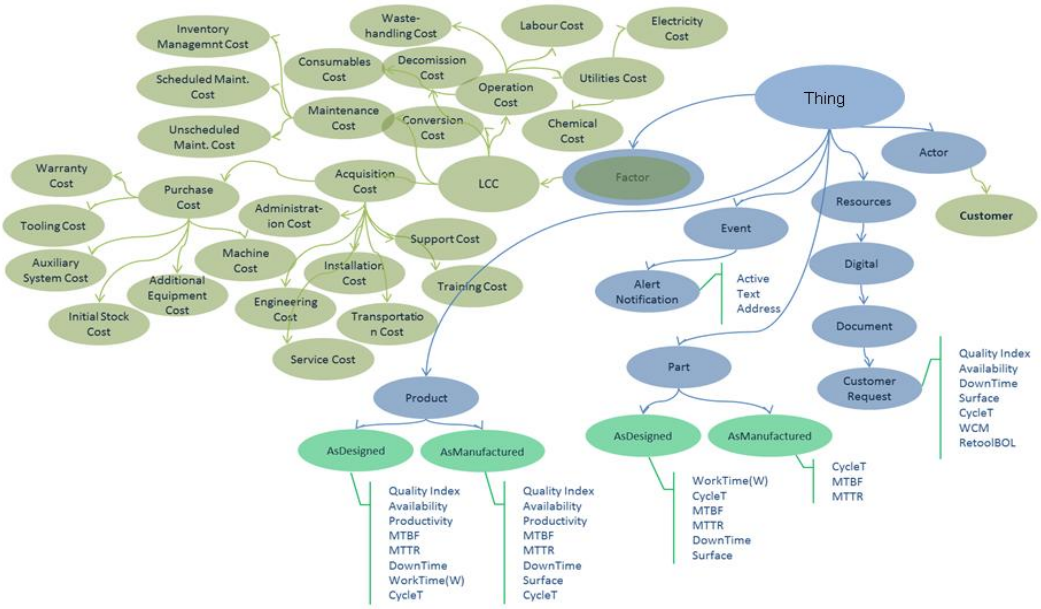


Figure 15 LCC ontology

In the proposal/concept phase, designers make decisions on the line configuration to be presented to the customer. Assembly lines are generally made up of many stations where a certain set of operations are performed either by an operator or by automatic equipment. The company uses a system that automatically finds the composition with the lowest LCC value, but it is not able to take more complex requests from the users under consideration. The schema of information flow is given in Figure 14. A list of potential requests is given as a list of properties of the customer's concept in the ontology.

6.2.2.2 Ontology rules for the use case

To enable advanced control of products design and maintenance, we have created three groups of rules:

(1) Rules for enforcing customers' requests. These rules control if the current solutions according to LCC minimization are in correspondence with customer requirements for product properties.

(2) Rules for inheritance of properties from part to product. These group of rules is used for automatic calculations of production line properties based on set of chosen stations. For example, surface of product is sum of part surfaces, while the productivity of a product is equal to lowest part productivity.

(3) Rules for alerts in MOL stage maintenance. These rules are used to alert service teams when the production line is not functioning as it was designed to, using automated data collection from the field.

SWRL implementation of all rules can be found in Appendix B of this document.

6.2.3 Use case on knowledge management

In the third use-case company, knowledge is stored and structured using K-brief methodology. Example of empty K-brief template is given in Figure 16.

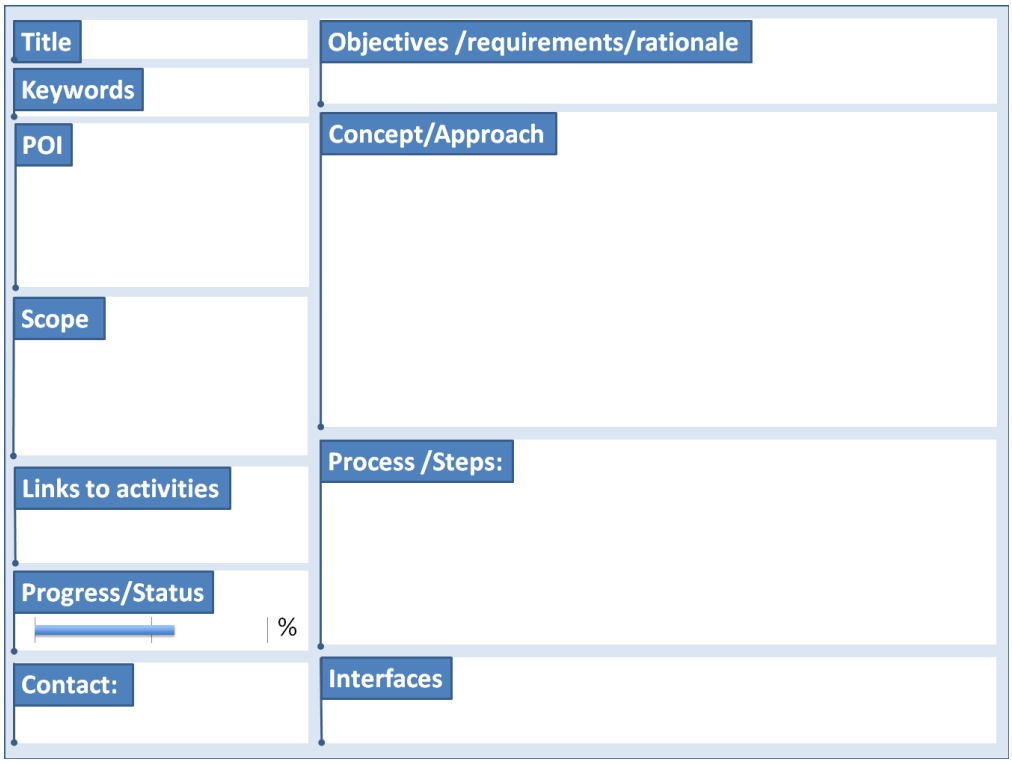


Figure 16 Example of K-brief as method for knowledge management

Employees capture their experience and work in digital forms, which are then exchanged, browsed and retrieved. K-briefs can number different content types. K-brief can hold notes from meetings, design solutions, error logs, industrial standards and regulations, employers profile, etc. To enable quick and high quality of information retrieval from this knowledge structure, k-briefs are tagged with categories such as author, project, product or discipline, it belongs to, as well as some other useful metadata.

6.2.3.1 Functional requirements for the use case

Ontology that models this domain is given in Figure 17. All the concepts are defined in upper PLM ontology but the domain is specific for the number of relations that were not present in that template.

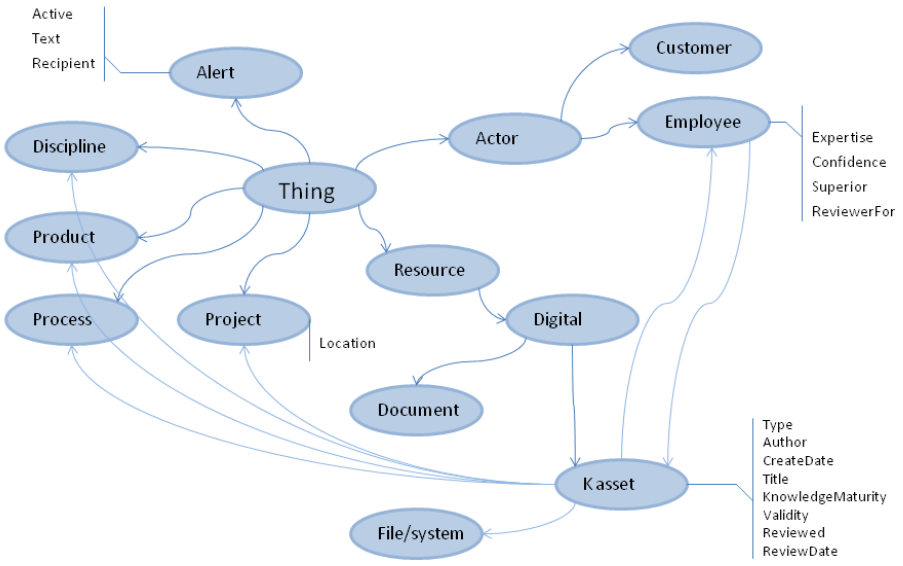


Figure 17 KM ontology

To be able to maintain a high quality of knowledge base, K-briefs need to be reviewed and validated and for this purpose, we introduced a property for every K-brief, named Maturity. The first requirement for knowledge management ontology is to be able to maintain the value of maturity for every K-brief. Second, to enable easy retrieval of knowledge, K-briefs are tagged with keywords, according to their content and tags need to be manually added once the K-brief is created. Considering that ontology contains relations between products, its parts and modules, the second requirement enables an automatic recommendation of additional key words, based on a set of existing ones. Further on, since the company has projects in different parts of the world, two same oil-drilling platforms will have to follow

different industrial standards depending on the location. For this reason, we use ontology to enable automatic industrial standard recommendation, based on project characteristics. Finally, as a contribution to this Dissertation, we propose a novel approach for industrial standards application. The idea is to model the standard in ontology, adding constrains in the form of concept properties and then using rules to control the designer's decisions. Within the scope of a project, we can only provide a methodology and an example of its application, considering that the number of standards used is very high.

6.2.3.2 Ontology rules for the use case

To enable these diverse functionalities, three groups of rules have been defined:

(1) Rules for maintenance of K-brief validity and maturity. Based on who and when reviewed them, K-brief can be assigned different maturity levels and this set of rules automates this process;
(2) Rules for recommending industrial standards appropriate for a project. Relevant industrial standards are recommended via alerting system, based on location, discipline and process;
(3) Rules for enforcing the application of semantically enriched standards. A special ontology is developed, representing semantically enriched industrial standards, which use alerting mechanism when a designer's decision is violating certain regulations.

Rule implementation of first and second group using SWRL is given in Appendix C.

6.3 Implementation

In this research, all mentioned ontologies and rules are implemented using Protege 4.2.0. It has a graphical user interface, and number of supporting plug-in developed in open-source community. As an example, Figure 18 shows implementation of Norsok concepts and Figure 19 shows implementation of LCC ontology with visual representation of ontology, generated by Ontograph [35].

[35] http://protegewiki.stanford.edu/wiki/OntoGraf

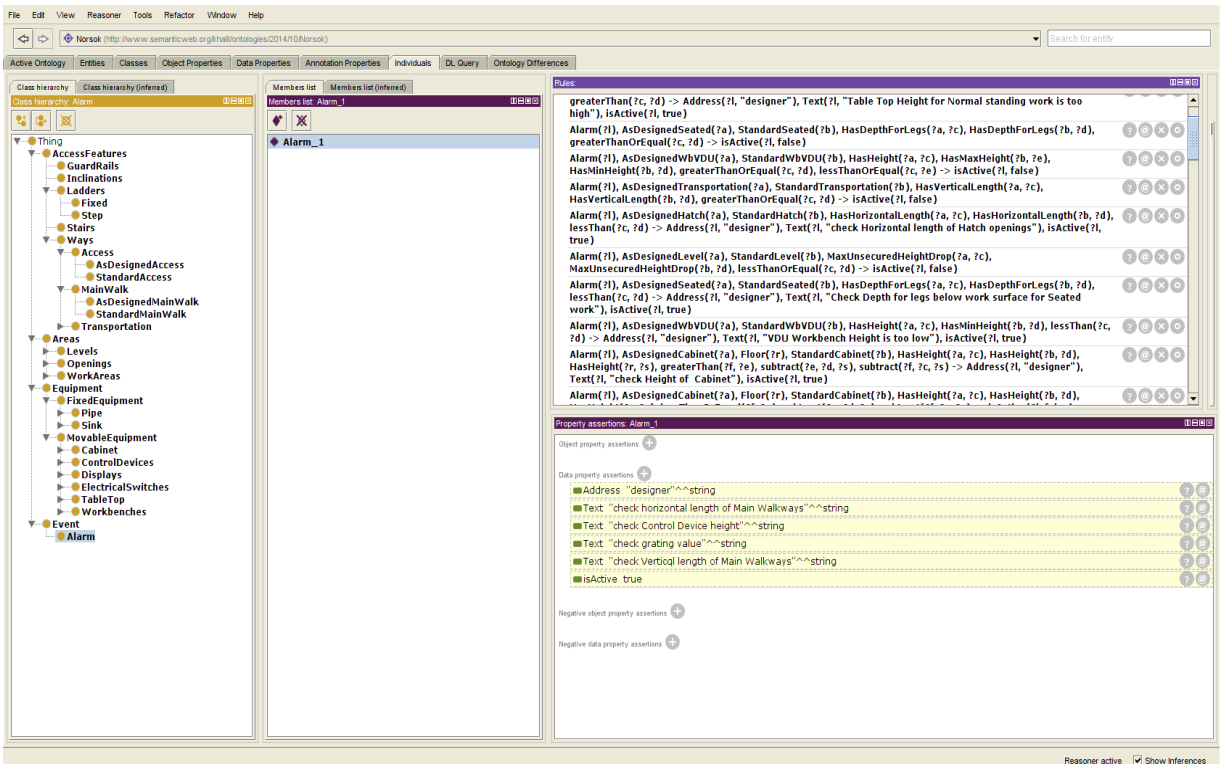


Figure 18 Norsok concepts in Protege

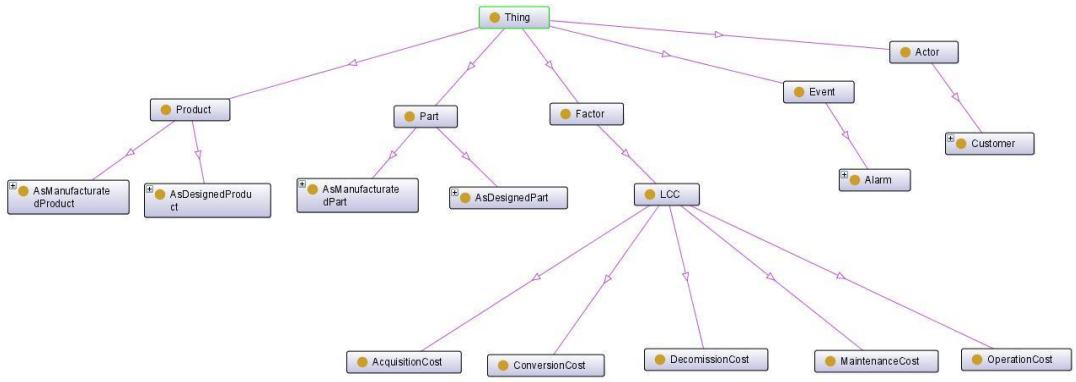


Figure 19 Ontograph of LCC in Protege

Following, the ontology rules for the same use-case are implemented by coding in SWRL language which is supported by Protege. Rule implementation, as well as results of inference by Pellet reasoner are presented in Figure 20.

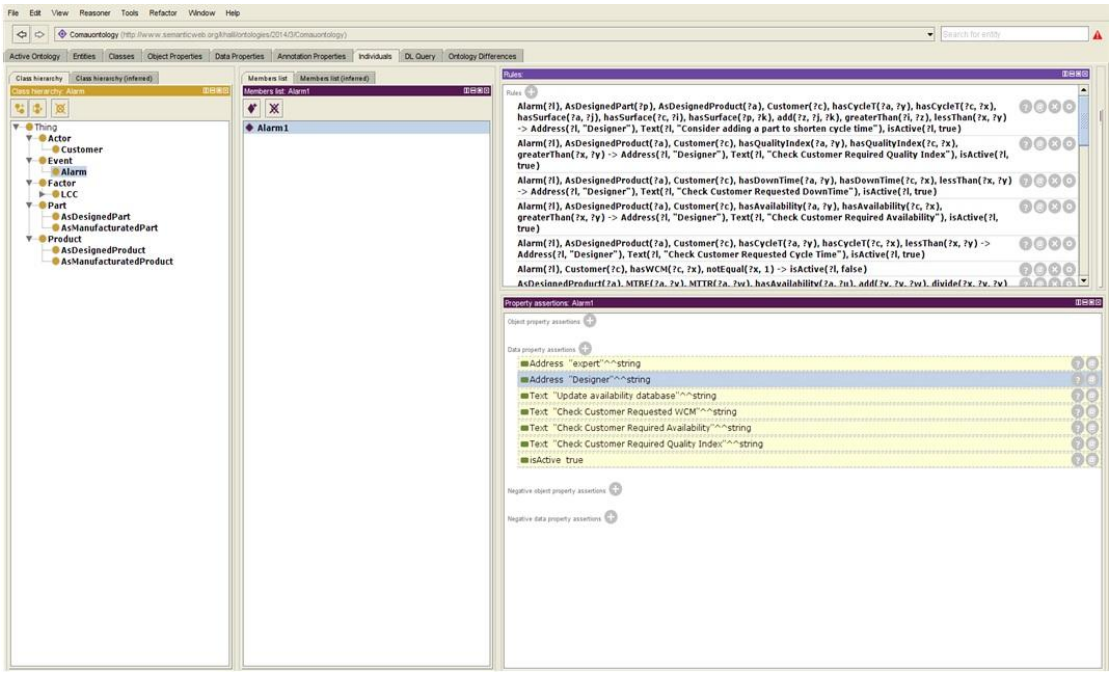


Figure 20 LCC reasoning in Protege

After the reasoning is complete and ontology is enriched with inferred attribute values and new relations, the next step is to extract related instances so that they be forwarded to a data mining module. As mentioned, in this research, data are extracted from ontology so that each data set contains instances from pair of concepts that are related in the ontology. We limited the distance to one relation but not including parent-child relations, as superconcepts are usually not populated. For each relation, a SWRL query is defined, so that it returns a pairs of instances exports them to CSV format. To implement this procedure, SWRLQeuryTab plug-in was used, although for this purpose an older version of Protege was used. Example of querying for data export is shown in Figure 21.

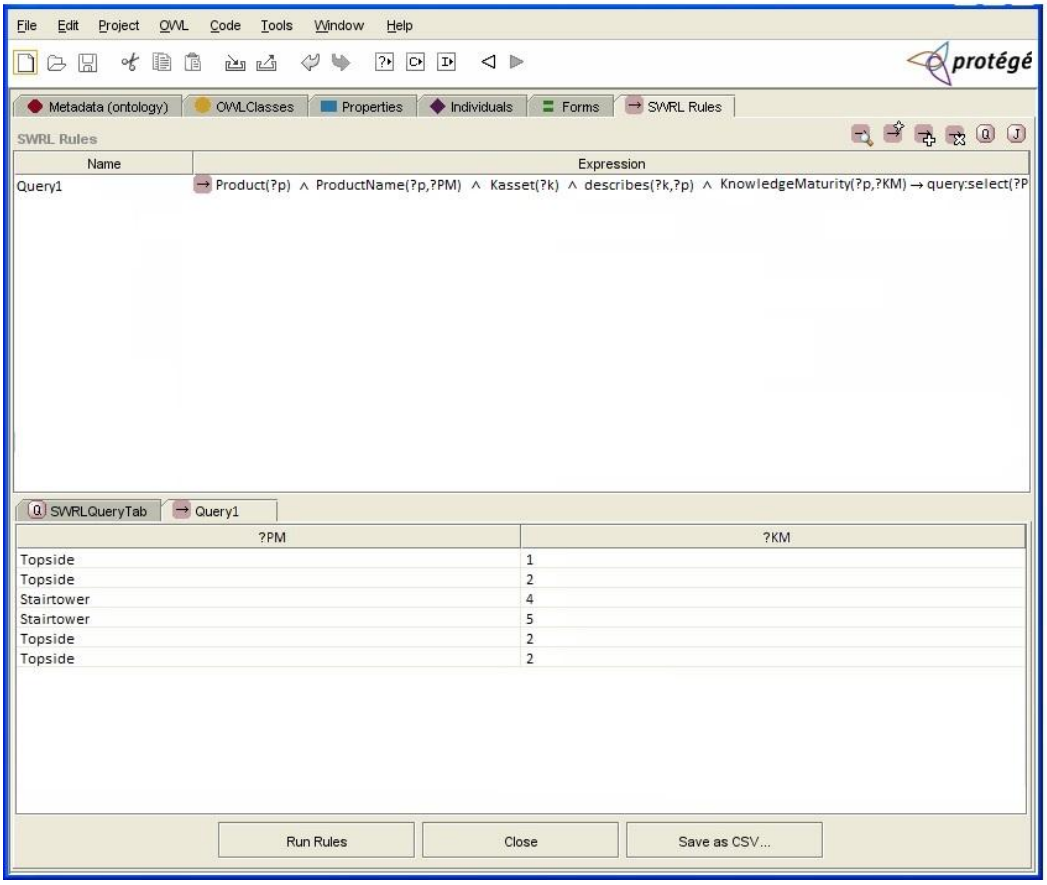


Figure 21 Data export from ontology in Protege

6.4 Conclusion

Through the exploitation of the user story mapping method for domain knowledge collection, ontology design time is reduced and the process is simplified. Upper PLM ontology was designed through a generalization process, so that it spans the entire PLM domain on an abstract level. Thus, upper ontology can be used as a template for the design of domain specific ontologies which lead to even shorter design times. It is important to note that ontologies, designed based on the same template, will share terminology and graph structure, which increases interoperability and simplifies ontology merger processes.

Following, to build on top of the static collection of knowledge, ontology rules are employed to develop non-standard and non-trivial functionalities of the semantic model. Developing such functionalities within a knowledge base, ensures that update errors or system inconsistencies are prevented.

Chapter 7 Data mining module definition

In this chapter, a method for overcoming obstacles of data mining automation is proposed in a context of PLM domain and for the purpose of detecting the correlations which are not assumed. A system presented which receives, as an input a complete, populated ontology and returns as an output, the correlations and patterns which exist within properties of the instances. Design and population techniques of such ontology are still an open research questions (Ana Milicic et al., 2013) but efficient solutions are proposed in previous chapters. The assumption that the system is used within PLM domain is relevant for several reasons. First of all, to design a PLM ontology, a semantic expert had to gather information about the domain, understand the key concepts and capture the dynamics of the domain. In a sense, this is very similar to the task of a data miner, when he is first starting to collect and form the data set to explore. The difference is that the data miner knows in advance which target variable he needs to model, while the ontology expert has to cover the entire domain with all its details and no specific perspective. Still, capturing the key concepts and defining their organizational structure performed by the ontology expert is significantly covering the task of data miners and tackling the first obstacle for data mining automation. The second motive for PLM ontology assumption is that, due to the large number of different actors and their expertise, PLM is expected to be a fruitful domain for detecting unforeseen co-dependencies between factors and at the same time, these dependencies are expected to bring a high added value in this domain. Finally, system efficiency relies on the assumption that the majority of patterns in the PLM domain are not highly complex. Under complexity of the pattern, we assume that the regression problems patterns can be estimated by polynomial functions of low degree and that classification problems patterns are easily separable clusters (as a contract to for example, chessboard classification test bench). This last assumption leads to a final important notion about the contribution of this research. It has been proven that the data mining process can never be fully automatic (Faloutsos & Megalooikonomou, 2007) in discovering the best fitting model. However, authors argue that the system which is fully automatic in discovering the underlying models with reduced precision is still an added value for the PLM community. For example, if the system can self-initiatively deduct that the manufacturing line has higher yield when row materials are

bought from supplier A than from supplier B, it is still an added value, no matter if the yield for 4% or 6%. Thus, in this research proposed is trading precision for the purpose of designing fully a automatic system which covers the entire PLM domain. The same argument, is used to justify the assumption about simplicity of the patterns whose detection is expected.. Even if some highly complex patterns are left unnoticed, every detected pattern creates an added value for the interested parties.

7.1 System architecture

The full system was designed following the procedure steps listed above as it can be seen in Figure 22: Overall system architecture. It includes additional module for straight forward correlation between attributes detection, since such results are naturally comprehendible to even absolute non-experts in data analysis field.

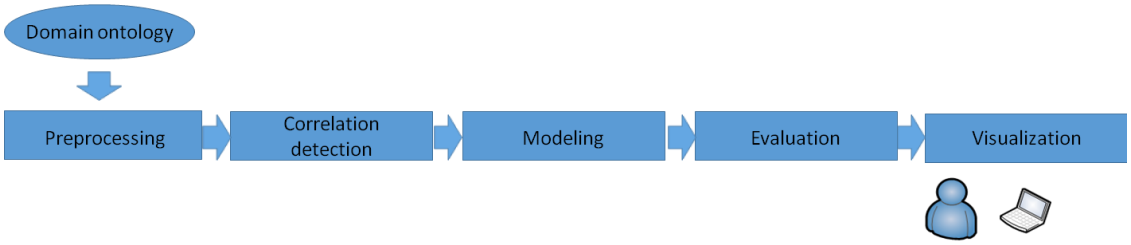


Figure 22: Overall system architecture

7.1.1 Data understanding

Data understanding is process where human expert needs to understand the domain from which the data are collected, how they are collected and how they are co-dependent. As previously argued, this task is very similar for a data miner collecting the data and an ontology expert, understanding the domain to be modelled. In both cases the main factors, correlations and variables, describing the topic of interest, need to be discovered and defined. At that stage, data miners work is finished, while the ontology expert still needs to systemize the knowledge into a coherent semantic structure. This means that the ontology carries additional information that is knowledge about the structure and relations between the objects of interest and that is a first key point of this chapter contribution. Proposed here is a data set export from the ontology, so that one data set combines the attributes of any two concepts that are related within the ontology. Such strategy will lead to a number of data sets that combine the information about the objects that are somehow dependent in real

life. Authors argue that in such data sets it is reasonable to expect to successfully mine for added value. Additional benefit from using ontology combined with proposed strategy for data set extraction, is that ontology has a built-in reasoning mechanisms that are inferring relations between concepts. Defining the base set of ontological rules and applying a first order logic on it will result in new relations, not present in the initial ontology design. Consequently, new rules will result in more data sets to be processed.

7.1.2 Data pre-processing module

As explained in the previous chapters, the data pre-processing step is the most challenging to automate. When handled by human experts, it will reflect their personal style and previous experience. In the same time, it is the critical point in expressing the contribution of our work presented in this chapter. Using the ontology, as a base layer of a data mining system is crucial for automation of pre-processing step. To emphasize the significance, the main obstacles in pre-processing are given and it is argued on how the data drawn from PLM ontology, handles these obstacles. The structure of this module is given in Figure 23.

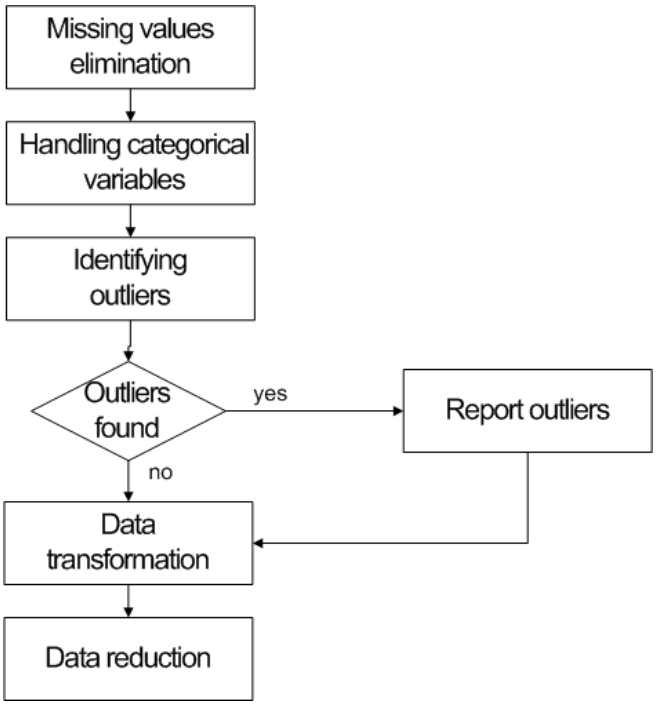


Figure 23: Pre-processing module structure

(1) Dealing with missing values: Since the ontology is mostly populated by different data sources such as sensor systems, counters and digital forms, missing values can be expected as a consequence of hardware dysfunction or human error. In both of the-

se cases, missing values are highly probable to be limited to several instances or several attributes. Since the purpose of the system is in detecting unforeseen patterns and correlations, the natural choice is to omit compromised instances and attributes as they have a low level weight as information carriers. Additional action to be considered in this step is the removal of attributes with no variance as they carry no useful information for the purpose of this system. Although damaged instances do not carry significant information for pattern detection, their existence still carries potentially relevant information for end-user. So, although those will be removed from the data set, their removal will be reported to an end-user.

(2) Dealing with categorical attributes: in order to handle the data set in any software environment, potential values of categorical attributes need to be coded in a programmable sense. One important characteristic of categorical data is that they do not have the distance measure defined (is red is more distant from yellow than from green?). This characteristic has to be preserved when coding, since many of the algorithm deal with distance values. One of the typical solutions (Hardy, 1993) is to use so called "dummy coding", for number of benefits that are outside the scope of this dissertation.

(3) Identify outliers: most of the environments for editing an ontology have tools for defining the attributes value types, ranges and potential values. Being so, it is expected that cases of inconsistent data (negative distance for example) are eliminated during ontology design. On the other hand, PLM domain is specific in a sense that the outliers are often a valuable piece of information. For example, too high temperature on a manufacturing line can indicate an error in the production setup. Another example would be detecting a fraud or unexpected losses in sales department. This means that once the outliers are detected in the pre-processing phase, they should not be simply discarded, but rather reported to human end-users. As there is no target variable known in advance, an empirical rule is used, that is a three standard deviations rule for continuous values with mean and deviation recalculation, while a histogram method is chosen for categorical attributes.

(4) Data transformation: A mandatory step in every data processing is data normalization, as most of the algorithms are highly sensitive to absolute values of attributes. Scaling by using mean and standard deviation is applied. Aggregation or generalization are omitted, since this optimization was already made in ontology design stage.

(5) Data reduction: In real life situations, the number of instances in the ontology will be naturally accumulated in higher or lower pace to a quite large numbers. This creates a necessity for a data sampling module. Although in real life, mostly normal distributions are expected, for generality, we choose rejection sampling method (Neal, 2007) and so that will always result in a 5 to 10 thousands instances. The data set size has been defined so that it can carry information on non-complex patterns that are aimed and not influence the time performance significantly. Further on, although all the attributes were estimated as relevant during the ontology design, a principle component analysis (PCA) module is used to limit the number of attributes.

7.1.3 Correlation detection

In the domain of PLM, it may come as very useful to have straightforward correlation between attributes detection. When the data sets are generated as previously explained, they will naturally contain attributes coming from different life cycle stages, with different actors involved. When the ontology is covering entire product life cycle, where no actor has access to all parts of the domain, it seems rational to expect that some valuable dependences are left unnoticed. At this stage, focus is on correlations between only two attributes, as even in this case, a variety of cases needs to be tackled and even more, number of challenges when it comes to result analysis. In general, statistic tests can be chosen for three possible cases:

(1) Continuous to continuous attributes : For this case, Pearson's coefficient for linear correlation (Pearson, 1895) is chosen. To increase flexibility, logarithmic transformation is applied and then repeat coefficient calculation. If any of these values, show to have an absolute value greater than 0.5, it is considered as relevant. This threshold is chosen empirically. The formula for calculating the coefficient is :

$$r = \frac{\sum_{i=1}^{N}(X_i - \bar{X})\,(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{N}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{N}(Y_i - \bar{Y})^2}}$$

Equation 1 Pearson's coefficient

where $\bar{X}$ and $\bar{Y}$ are mean values of two variables in question

(2) Continuous to categorical attributes: Based on which attribute is declared as dependent, two different cases of problem can be defined but as there are only two variables , we chose a simpler method. The quality of clustering of the continuous variable by category membership using Davies-Bouldin index is estimated (Davies & Bouldin, 1979). Again, the empirical threshold for relevancy recognition is to be ad-

justed. If the C_i is one of the clusters, and T_i and A_i are its size and centroid, than its measure of scatter within the cluster is :

$$S_i = \frac{1}{T_i}\sum_{j=1}^{T_i}\left\|X_j - A_i\right\|_2$$

For the measure of separation of cluster C_i and C_j where $a_{k,i}$ is k-th element of A_i we have :

$M_{i,j} = \sqrt{\sum_{k=1}^{n}\left(a_{k,i} - a_{k,j}\right)^2}$ and following, the measure of how good the clustering is : $R_{i,j} = \frac{S_i - S_j}{M_{i,j}}$

Now we can define the measure of quality of each cluster as $D_i = \max_{j:j\neq i} R_{i,j}$

Finally the DB index can be calculated as for N number of cluster

$$DB = \frac{1}{N}\sum_{i=1}^{N} D_i$$

Equation 2 Davies-Boulding index

(3) Categorical to categorical attribute : to estimate the dependency between membership to a category of one variable to membership of corresponding category in second variable, slightly adjusted Pearson's chi-squared test (Plackett, 1983) is chosen, as the most commonly used one for similar problem setups. The idea is to establish whether difference between sets of data clustered by values of one variable is likely to arise by chance. So for each category of one variable, we determine whether there are significant outliers of co-occurrence frequency among categories of second variable. Let's say we have N points with two categorical features, which have p and q numbers of different values. Then for each value of first feature, we can estimate whether it has frequent co-occurrence with values of second feature using formula

$$\theta_{i,j} = \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}$$

Equation 3 Pearson's chi-squared test

where $O_{i,j}$ is number of observed co-occurrences of two values and $E_{i,j}$ is expected number of co-occurrences according to uniform distribution. After calculating these values for every pair of values for two variables, the significant conclusions are determined by outlier detection procedures.

Now that a numerical qualitative measure for the level of correlation between the data is selected, the next step is to determine the values above which these results are relevant enough to be presented to the user. Using the term "relevant" might be a bit vague, so an attempt to explain it more closely is given. The purpose of the system presented in this dissertation is to detect unforeseen patterns and correlations in automatic, self-initiated manner. As such, it will be reporting results to an end user who might be very familiar with the domain, or very unaware of patterns present in the domain. For example, a good domain expert might be overwhelmed if the system is reporting all the strong (and thus expected) correlations with index close to 1. He might be more interested in a less obvious connection and he would chose not to have the obvious ones presented to him. This leads us to a term "relevancy" of the results found, for a specific user. Initial thresholds of relevancy (correlation indexes), above which the results are presented to the user, can be empirically selected, but the adjustments have to made based on a user feedback.

7.1.4 Modelling

First important factor when it comes to modelling the data sets in this systems is that time efficiency is not a significant constraint. Since the system is automatic and self-initiated, it can be thought of as a back engine, that can be left running and storing its results. As a consequence, a number of modeling attempts can be performed, even the ones that will never be communicated to the end user. Having this in mind, the obvious question is tackled, that is, what is the purpose of the model as there is no known target variable. Following previously explained setup, the answer is that sequentially, one by one variable from the data set is declared as target and then attempt to model it is made, using the remaining data. Many of these attempts will show no results, but as long as one or few does, we have achieved a gained value.

Every data modeling procedure starts with data splitting into training, validation and testing subsets. As usually, training data is used for learning the model, validation for parameter tuning and testing data is for final model estimation. Typical values of 70%-15%-15% split are chosen, except in cases where there are less than 1000 instances. In this situation, k-fold method is evoked (specifically 5-fold). In extreme cases where there are very few instances even for this method, the simple conclusion is that the data is not sufficient for reliable conclusions. It is important to notice, that tackled here is the challenge of selecting not only the model but also the modeling algorithm, so the same data subsets defined in this step have to be used throughout all of the trials.

When it comes to selecting a modeling algorithm to be used, it is again an example of decision making process where human expert displays his experience and style, characteristics which digital systems cannot display. On the other hand, automated system has the advantage of time and processing speed. The problem was tackled so that the advantages are maximally exploited. Namely, through extensive literature research, group of algorithms are selected, for both classification and regression problems, that are most simple and with low number of parameters to tune, noise resistant and most often used to model the real life data with low complexity of patterns. What processing speed brings is possibility to use every algorithm from the group on every data set and only in the end, based on test data set, the best algorithm is selected, together with the best model. For this reason, the algorithms selected are such that they complement each other in advantages and limitations as well. Next, listed is the group of algorithms with brief descriptions their advantages. First listed are three classification algorithms and followed by three regression algorithm. It is important to note that the classification problems with multiclass target variables (n values) are decomposed into n binary classification problems which is a standard approach in such problem setups.

7.1.4.1 Decision tree

The decision tree is graph-like model where each internal node represents a cut-off value for one of the variables and leafs are target variable classes (Rokach, 2008) . Higher nodes carry more weight. In this system decision tree is used as classification algorithm and it is one of the most popular methods in data mining community. This algorithm was selected for number of reasons. First of all, it is non-parametric, which makes it more attractive for automatic training. Further, it handles continuous and categorical attributes equally well which is not a very frequent quality. Finally, its training is performed in a reasonable time. In the literature, the down-sides of decision tree are namely that its training techniques do not guarantee optimality, which is in our case, as previously explained, not a priority. One true problem remains over-fitting in a case of very noisy data. Here we focus on a particular tree-based framework called *classification and regression trees*, or *CART* (Breiman, Friedman, Stone, & Olshen, 1984), although there are many other variants going by such names as ID3 and C4.5 (Quinlan, 1986, 1993) . The simple representation of training process is given in Figure 24.

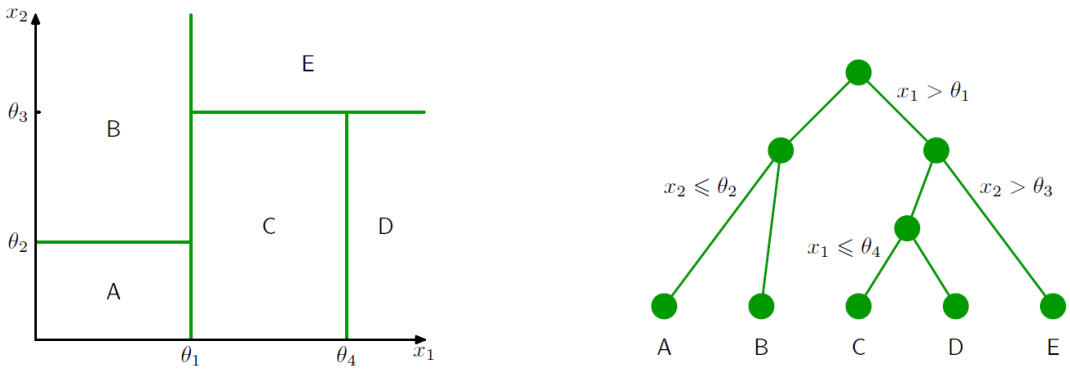


Figure 24 Classification tree training

In this example, the first step is divides the whole of the input space into two regions according to whether $x_1 < \theta_1$ or $x_1 > \theta_1$ where θ_1 is a parameter of the model. This creates two subregions, each of which can then be subdivided independently. In order to learn such a model from a training set, we have to determine the structure of the tree, including which input variable is chosen at each node to form the split criterion as well as the value of the threshold parameter θ_i for the split. The task is also to determine the values of the predictive variable within each region. To solve this problem, greedy optimization is applied, starting with single root node and then growing the tree by adding nodes one at a time. At each step there will be some number of candidate regions in input space that can be split. For each of these, there is a choice of which of the input variables to split, as well as the value of the threshold. The joint optimization of the choice of region to split, and the choice of input variable and threshold, can be done efficiently by exhaustive search noting that, for a given choice of split variable and threshold, the optimal choice of predictive variable is given by the local average of the data. The next issue that imposes is when to stop growing tree. It is found empirically that often none of the available splits produces a significant reduction in error, and yet after several more splits a substantial error reduction is found. For this reason, it is common practice to grow a large tree, using a stopping criterion based on the number of data points associated with the leaf nodes, and then prune back the resulting tree. The pruning is based on a criterion that balances residual error against a measure of model complexity. For greedy optimization, measure of performance used is cross-entropy, since it is more sensitive to node probability and unlike misclassification error, it is differentiable and thus better choice for gradient based optimization methods.

7.1.4.2 K-nearest neighbor

Second classification algorithm selected is K-NN where the class of the instance from a validation set is selected based on majority voting of classes from k closest instances from training set (Altman, 1992). This algorithm is non-parametric and the only choice for human ex-

pert would be the value of k. This choice, although empirical, strongly effects the performance of the algorithm. In this case again, the advantages of an automatic system are exploited to allow trials for different values of k, specifically from 1 to 5 for binary classification, or up to number of classes in multi-class problems. In general, higher k performs better on noisy data in a case where we expect a relatively simple patterns as we do in a case of PLM domain. Strong side of K-NN is that it has guaranteed worse error rate and it is very simple to train. Another benefit is that with proper choice of value k, it becomes very resistant to noisy data. Finally, it handles a complex patterns very well. The down side is that it does not perform well on high-dimensional data, or high number of categorical data, simply because it is based on a distance measure. Although a dimensionality reduction step is performed during pre-processing, this will certainly remain a problem for some data sets. The general idea behind the algorithm is given in (Cover & Hart, 1967) Figure 25. After k parameter has been determined, there is practically no algorithm training stage, and the classification is performed by greedy search for nearest neighbor(s).

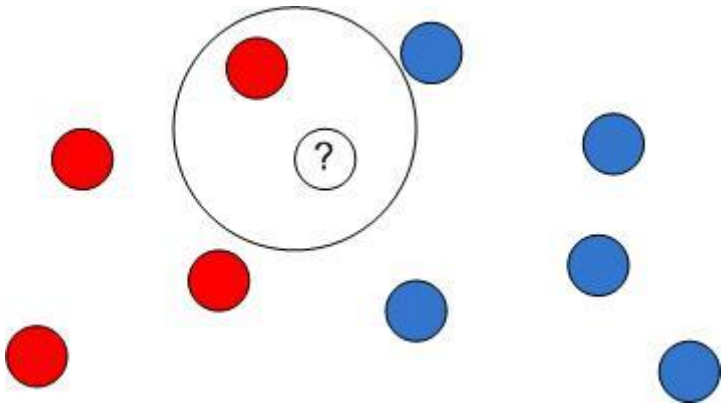


Figure 25 K-nn principle

7.1.4.3 Support vector machine

SVM was selected as part of the group to compensate for downsides of previous two algorithms. Most importantly, SVM is very good in handling high-dimensional data, even in cases when number of dimensions is higher than number of training instances (Burges, 1998). For this system, SVM with slack variable is chosen, to increase robustness to noise in data. Also a kernel trick with Gaussian Radial Basis Function (RBF) kernel is used to handle data which are not linearly separable. The pair of parameters, soft margin parameter and kernel width are selected using trials on validation set for several different values. Kernel width is kept close to empirical value of $1/\sqrt{N}$, where N is number of instances, while the soft margin parameter is gradually increased.

Support vector machine are based on two key ideas that can address a non-linear discrimination problem, and to reformulate the classification problem as a quadratic optimization problem. In the case of linear problem, we only use the notion of maximum margin. The

margin is the distance between the boundary of separation and the nearest samples. These are called support vectors. In SVM, the boundary separation is chosen as the one that maximizes the margin. The problem of finding the optimal separating boundary, from a training set Figure 26. is solved by formulating the problem as a quadratic optimization problem, for which there are known algorithms. In this system, Langrangian optimization is used

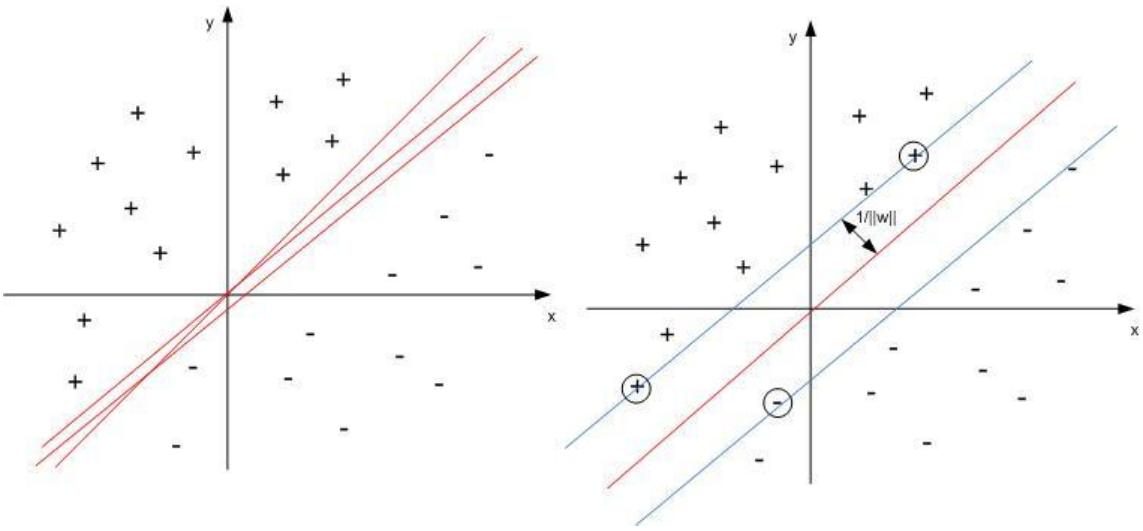


Figure 26 Maximum margin for SVM

Entire solution of dual problem is outside the scope of this dissertation, but the detail can be found in many textbooks (Vapnik, 2000).

Further on, to tangle the problem of non-linear separating bound , we use kernel method, which is very often used in combination with SVM. Kernel methods owe their name to the use of kernel functions, which enable them to operate in a high-dimensional, *implicit* feature space without ever computing the coordinates of the data in that space, but instead, by simply computing the inner products between the images of all pairs of data in the feature space. This operation is often computationally cheaper than the explicit computation of the coordinates. For this system, we use one of the most popular kernel functions, that is RBF function :

$$k(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right)$$

Equation 4 RBF kernel

where x are data in feature space and γ is kernel width. Without going into details which can be found in (Schölkopf & Smola, 2002), the main principle is presented in Figure 27[36]

[36] http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html

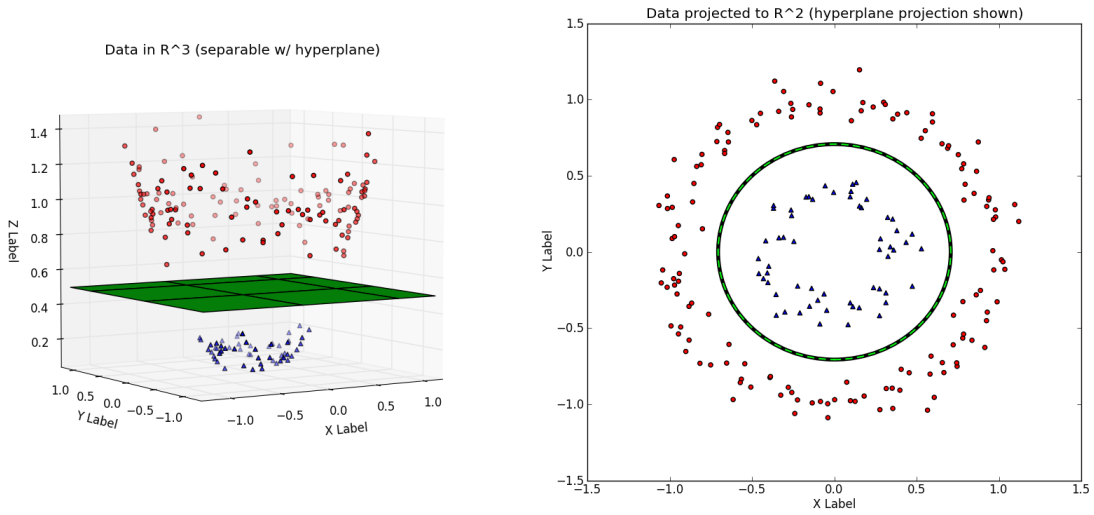


Figure 27 Effect of kernel trick

7.1.4.4 Linear regression model

As a first algorithm for regression, linear regression model is chosen as it is the most straight forward model with well-established techniques for training and handling different limitations (Bishop, 2006). For training, least squares method is used which has a convenient closed form solution. The benefit of this algorithm is its simplicity and robustness in a case of Gaussian noise. The downside of this algorithm is that it is very much sensitive to a presence of outliers. Although this issue is addressed during pre-processing step, another precaution measure is added in a form of distance-based approach for outlier detection. Specifically, we employ simple k-NN-distance model. For this system, due to previously explained assumptions about the domain, we limit basis function selection to linear function. The task is very much straightforward, that is, given the set of training points, find the linear function:

$$y(X, w) = Xw + \beta$$

Equation 5 Linear regression model

that captures the underlying generating function in the most accurate manner. Here X_{nxm} is matrix of training data, with *n* points of dimensionality *m*, *w* is regression coefficient vector, β is noise vector and *y* is a prediction vector. Example of the simplest case when dimensionality of input space is 1 is given in Figure 28.

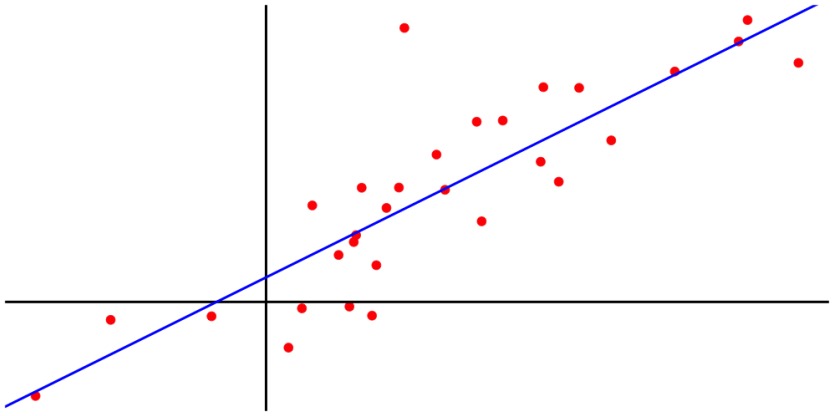


Figure 28 Linear regression

For estimating the model we use least-square error estimation which has to be minimized and which has a convenient closed-form solution.

$$S(w) = \sum_{i=1}^{n} (t_i - x_i^T w)^2 = (t - Xw)^T (t - Xw)$$

$$\hat{w} = (X^T X)^{-1} X^T t$$

Equation 6 Closed form solution for linear regression

7.1.4.5 Interpolation

Interpolation is chosen as a tool for estimating non-linear functions in a case when small level of noise is present in the data. It is also often efficient when variables have different behavior on different interval. Specifically, spline interpolation is used with a polynomial of third degree to enable extrapolation. Cubic spline (McKinley & Levine, 1998) can handle well several dimensions but in a case of high-dimensional data, interpolation might have degraded efficiency. In that case, the choice goes to K-NN interpolation, which will never be the most accurate but for some data sets, remains as the best alternative.

The idea behind spline interpolation is that error extrapolation might be lowered if we model polynomial functions piece-wise in the interval, instead of the entire range. So, the range of independent variable is split with equidistant points and function between every two sequential breakpoints with a condition that at every knot, two functions must merge in smoothly so that the first and second derivatives at those points are continuous. Example of benefits of using this specific method are given in Figure 29.

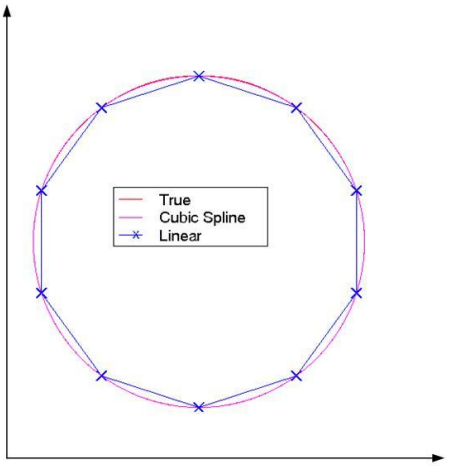


Figure 29 Spline interpolation

For every feature dimension, extrapolation is done for every point from test set and the final value of prediction is determined as averaged value of all dimensions. Although there are more refined methods for handling multivariate functions (Gordon, 1971), it was determined that this approach satisfies requirements of PLM domain. Still when the number of dimensions excides 10, K-nearest neighbor method is applied. The principle is same as the one already described as classification algorithm, with one difference, that is, the test point are adopting the continuous value as target prediction instead of class label.

7.1.4.6 Polynomial curve fitting

For this most challenging task, the problem is limited to one-dimensional problems (Arlinghaus, 1994). Continues variables are modeled to target variable one by one using polynomial functions of second order. Using more complex models would require a human intervention and as previously explained, accuracy in this system is not the main priority. The method is summarized by the task of fitting polynomial function

$$y(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 .. + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Equation 7 Polynomial curve fitting

where M is the order of polynomial. This can be done by minimizing an error function that measures misfit between true and predicted value on training set data points. As an error function, mean square error is chosen (MSE). The choice of M remains an open question since a too low order of polynomial will result in poor estimation of original function, while too high order will result in over fitting and picking up all the noise present in the data as useful model information. The effect of M choice are given in Figure 30 (Bishop, 2006)

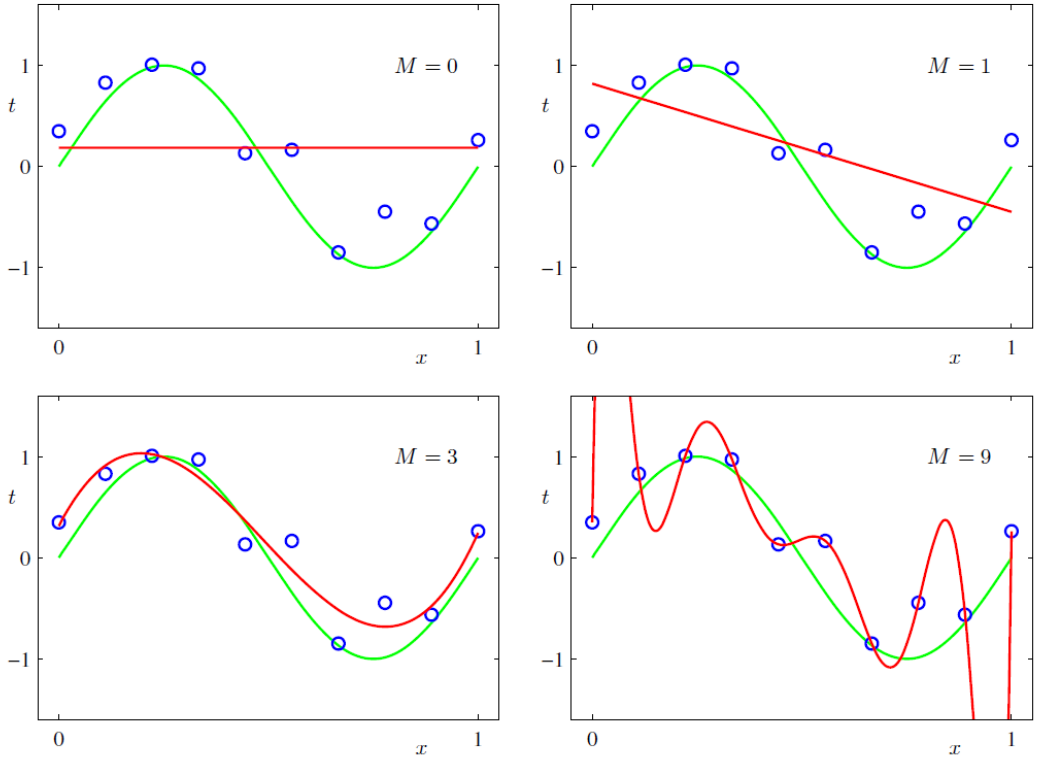


Figure 30 Examples of curve fitting with different choice of polynomial order

To prevent this, instead of minimizing only MSE, we introduced method called ridge regression where fitting is solved by minimizing following function:

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, w) - t_n\}^2 + \frac{q}{2}\|w\|^2$$

Equation 8 Ridge regression

The parameter q that is introduced controls complexity of the model and prevents over fitting. The important feature, although we will not give details of derivation here, is that this optimization problem has a close form solution (Hoerl & Kennard, 2012).

7.1.4.7 Support vectors for regression

The advantages and problems of support vectors when applied for regression are basically the same as when applied for classification. The concept and training procedure are the same, accept for the error function that is used. In this particular work, epsilon SVR is used and values of epsilon parameter is found by cross-validation. To enable capturing of non-linearity, previously described RBF kernel is applied. The graphical representation of concept is given in Figure 31.

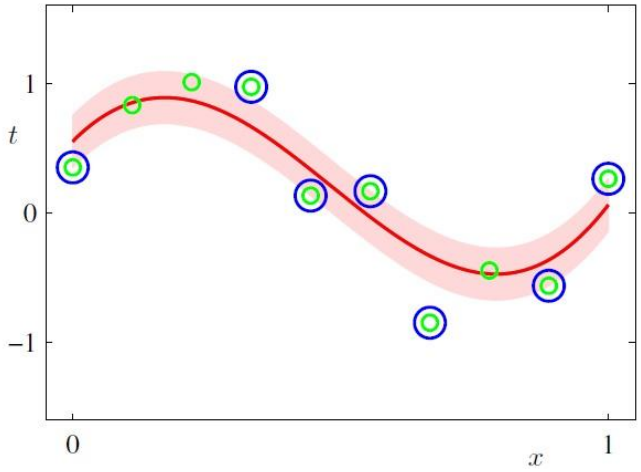


Figure 31 Support vectors for regression

Support vectors on an edge of epsilon-insensitive tube are marked in blue circles. This tube comes from the definition of error function which is minimized when training the model and in this particular case it has form :

$$E(y(x) - t) = \begin{cases} 0, & if\ |y(x) - t| < \varepsilon \\ |y(x) - t| - \varepsilon, & otherwise \end{cases}$$

Equation 9 Epsilon-insensitive error function

Details of the training process and application of this algorithm can be found in (Basak, Pal, & Patranabis, 2007)

7.1.5 Result evaluation

Three algorithms have been selected for each type of problem, classification and regression in such way that they cover a wide range of data set characteristics. Examples of covered sets are those with existing outliers, noisy data and high-dimensional data. Human expert would be able to assume which of listed algorithms will perform the best, based on these characteristics of the data, prior to model training, but the automatic system described, does not have such abilities. This means that results of each algorithm training and number of models created, have to be evaluated using the same criteria and only then, the solution can be selected. There are number of model evaluation methods in the literature all of which are recommended for specific types of problem. Considering that we cannot know in advance the type of problem in question, safe choice is to apply Occam Razor and select the simplest methods, which are usually the one most widely applicable.

For the classification problems, following the trends, Matthews correlation coefficient (Matthews, 1975) is used as it is known as the best summarization of confusion matrix. The formula for it is:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN) + (TN + FP)(TN + FN)}}$$

Equation 10 Matthews correlation coefficient

where TP is the number of true positives, TN the number of true negatives, FP the number of false positives and FN the number of false negatives coming from confusion matrix. The formula it returns a value between −1 and +1. A coefficient of +1 represents a perfect prediction, 0 no better than random prediction and −1 indicates total disagreement between prediction and observation. Thus, the models coming from all three classification algorithms are evaluated on test set data and the one with the highest MCC coefficient is chosen to be presented to the user.

For the regression problems already mentioned mean square error is used. The disadvantage of this method is that it is heavily weighting outliers. Still in this application it comes as natural choice since all algorithms and their models are evaluated on the same test data set and the outliers were removed. The model with the lowest mean square error is chosen as the one to be presented to the end user. The architecture of this module is given in Figure 32.

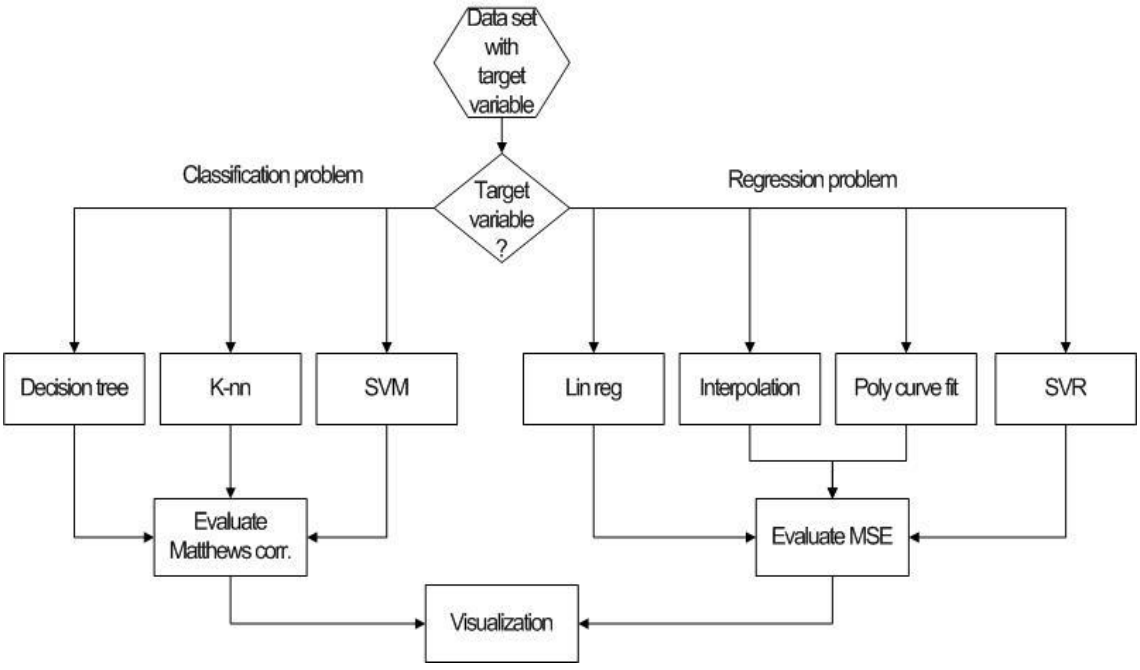


Figure 32: Structure of data modelling procedure

7.1.6 Results presentation

The question of visualization becomes a non-trivial issue when it comes to presenting the results of un-asked questions. Some of the algorithms have easy-to-understand representation, such as decision tree, while some are very difficult to grasp in general and even more in graphical sense. Most of the algorithms that even could be plotted in 2D or 3D cases, like linear regression, become complex when it comes to plotting multidimensional data. Depending on a algorithm that shows to perform the best on given data set, visualization technique will change and in many cases, they will not be useful. It is further complicated by the fact that there is no information about end-users level of expertise.

To ensure that the system acts appropriately in every situation, safe choice is that the default mode is the one for absolute non-experts in a field of data analysis. For the categorical variables, system communicates findings in a form of written text, such as "The attribute X can be predicted with t% confidence, based on [y_1....y_n] attributes". Naturally, such statements are generated only when t% is higher than the trivial solution of classification. Users who become interested, can explore further the details of the model behind this result. For the regression problems, the challenge is even more complex. The data after preprocessing are not intuitively understandable to end-users anymore and they need to be communicated using natural language. One of the proposals is to calculate mean of residuals in the test data set and, after reverse of normalization used during preprocessing, express it as expected absolute error of prediction. The statement would be "The variable X can be predicted with ±t error". This would be far from perfect interpretation of the result but it opens a door for further exploration of the model in a case that end-user is interested in modeled attribute. Finally pair wise correlation and outliers detected are also communicated in naturally formed statements.

7.2 Implementation

The entire data mining module was implemented using Matlab Figure 33. Exchange of data sets between Protege and Matlab is enabled by simply using Excel file with predefined name. Beside standard Matlab mathematical functions set, additional toolboxes involved are:

(1) Statistics toolbox
(2) Bioinformatics toolbox
(3) Optimization toolbox

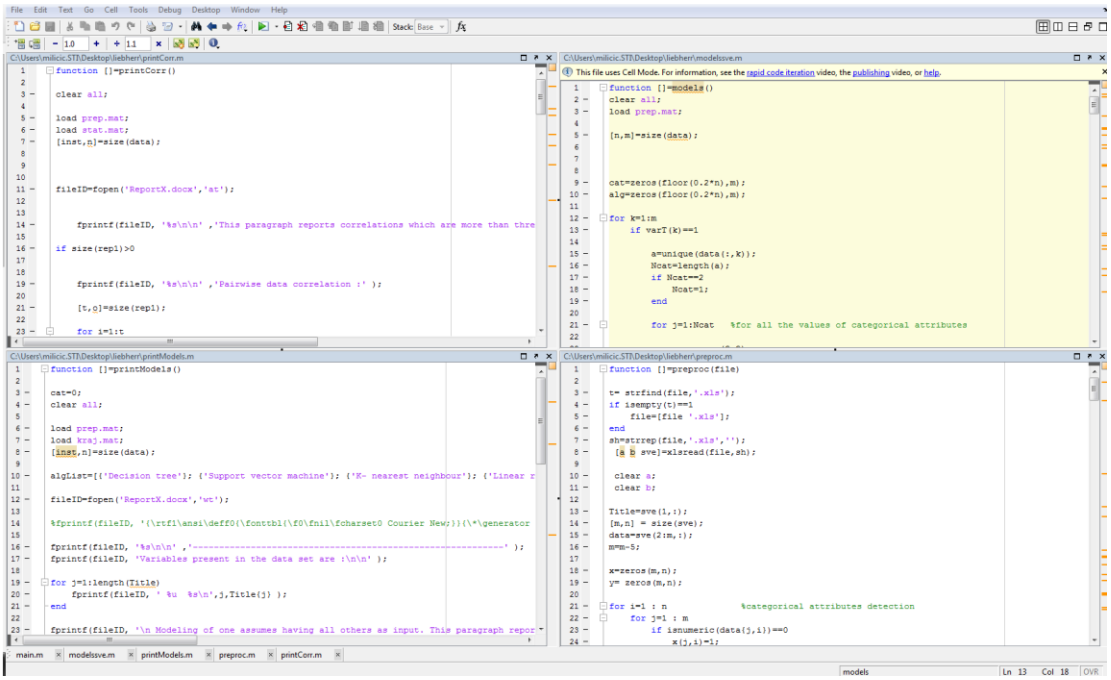


Figure 33 Matlab interface

Although final report contains only human-readable, easy to comprehend information, all the models and hyper-parameter settings are stored so that the expert user can access them and explore in more details. All the codes can be found in Appendix of this dissertation.

7.3 Conclusion

It has been shown in previous studies that the data mining procedure cannot be handled by a computer in a search of an optimal solution, at least not with any acceptable amount of memory and time resources. In this research, the data mining procedure is disassembled into the most basic steps and challenges and presented are the strategies for tackling those, based on relaxed time constraints and gain of sub-optimal accuracy assumptions. The system is fully PLM specific domain independent and assumed to be used by non-experts in a field of data analysis. As such, it promises an added value to all interested actors in the PLM domain. It is inevitable that the data exploitation tasks will become automated processes and authors argue that ontology is a good choice of a ground layer for such systems.

Chapter 8 Data mining module evaluation

For testing of the data mining engine (DME), number of publically available data sets was selected. These data sets are well known in machine learning community and they are used as benchmarks for estimating new learning algorithms. The results of testing are reported and stored for reference. Most popular data sets can be found in Machine Learning Repository [37]. These data sets have a predetermined target variable but since in our system, all variables are treated as targets sequentially, target variables were merged with the attributes and entire data set was used as input to data mining module. Report will contain models for all variables that are above accuracy threshold, including the real target variables' model. This means that we can compare performance of the best manually tuned algorithms to fully automatic DME designed in this research.

In total six data sets were selected, so that they represent a different real-life domains. This feature is significant since the estimation of outlier detection and correlation detection is as significant for this system, as the final accuracy on target variable modeling is.

8.1.1 Iris dataset

This is perhaps the most referenced data set in the pattern recognition literature. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other two while the latter are not linearly separable from each other. Some general characteristics are given in Table 4.

Data set	Attributes	Task	#Instances	#Attributes	#Web hits
Multivariate	Real	Classification	150	4	659802

Table 4 Iris data set characteristics[38]

[37] http://archive.ics.uci.edu/ml/
[38] http://archive.ics.uci.edu/ml/datasets/Iris

After loading the data in DME, automatically generated report, without any human intervening is given in Figure 34.

Variables present in the data set are :

1 Sepal length

2 Sepal width

3 Petal length

4 Petal width

5 Species

Variable Sepal length can be predicted with average of 0.26 relative mistake using Linear regression algorithm

Variable Petal length can be predicted with average of 0.12 relative mistake using Linear regression algorithm

Variable Petal width can be predicted with average of 0.15 relative mistake using Linear regression algorithm

---

Value "Iris-setosa" of variable "Species" can be predicted with 1.00 accuracy using Decision tree algorithm

Value "Iris-versicolor" of variable "Species" can be predicted with 0.97 accuracy using Decision tree algorithm

Value "Iris-virginica" of variable "Species" can be predicted with 1.00 accuracy using Decision tree algorithm

---

This paragraph reports correlations which are more than three standard deviations higher than mean of all correlations

Pair wise data correlation :

"Sepal length" and "Petal length" show correlation of 0.87

"Petal length" and "Petal width" show correlation of 0.96

Significant correlation between continuous and categorical variables were not found

---

This paragraph reports outliers in continuous variables :

Total of 1 instances has variable "Sepal width" with extremely high value of more than 4.37

Figure 34 Iris data set report

As mentioned, in the data set one class of iris specie is separable from the two remaining ones and accordingly DME has found that Iris-setosa can be predicted with 1.00 accuracy (Zhong & Fukushima, 2007). The other two classes are separable after certain non-linear transformations, which DME managed to capture but not to a full precision. No significant

correlations between attributes were found and there is one outlier with very high values of sepal width.

Although the dataset is classification challenge where Species is a target variable, in PLM domain when analyzing the data for unknown patterns, target variable is often unknown as well. For this reason, as previously explained, all variables are sequentially treated as targets. In this example, DME found that regression for three variables also gives significant accuracies, which is in compliance with the correlations found in pair wise testing.

8.1.2 Wine data set

These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each wines and to goal of original data set is to be able to predict the cultivar based on wine chemical attributes. Test characteristics are listed in

Data set	Attributes	Task	#Instances	#Attributes	#Web hits
Multivariate	Real	Classification	178	13	390183

Table 5 Wine data set characteristics[39]

Report generated for this data set is given in Figure 35.

[39] http://archive.ics.uci.edu/ml/datasets/Wine

Variables present in the data set are :

1 Family

2 Alcohol

3 Malice acid

4 Ash

5 Alkalinity

6 Mg

7 Phenols total

8 Flavanoids

9 Non-flav phenols

10 Proanthocy anins

11 Color

12 Hue

13 OD280

14 Proline

Value "1" of variable "Family" can be predicted with 1.00 accuracy using Decision tree algorithm

Value "2" of variable "Family" can be predicted with 0.98 accuracy using Decision tree algorithm

Value "3" of variable "Family" can be predicted with 0.95 accuracy using Support vector machine algorithm

Variable Flavanoids can be predicted with average of 0.26 relative mistake using Polynomial curve fitting algorithm

---

This paragraph reports correlations which are more than three standard deviations higher than mean of all correlations

Pair wise data correlation :

"Phenols total" and "Flavanoids" show correlation of 0.86

Significant correlation between continuous and categorical variables were not found

---

This paragraph reports outliers in continuous variables :

Total of 1 instances has variable "Malice acid" with extremely high value of more than 5.52

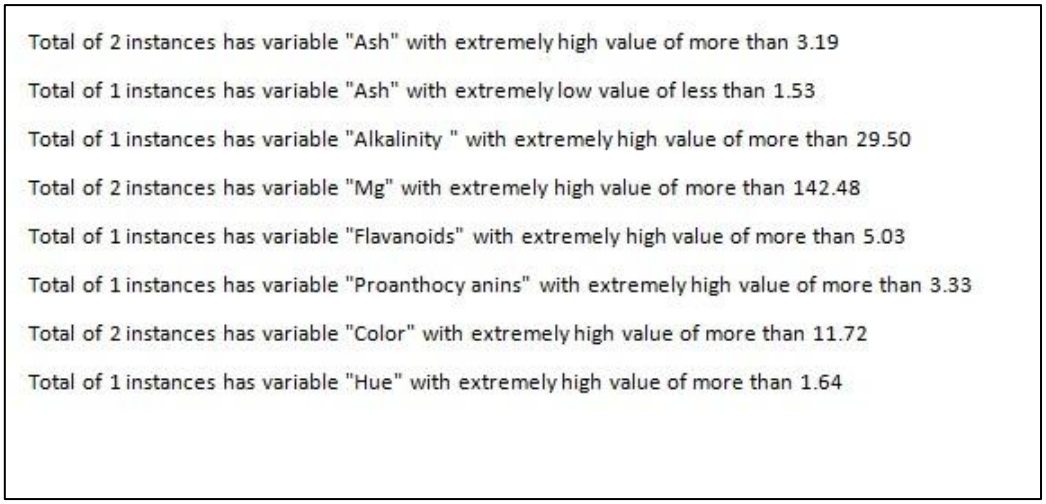


Figure 35 Wine data set report

The highest reported accuracy for this data set is that all three classes are separable (Tan & Dowe, 2005). DME has slightly lower accuracy, but as discussed, this is a trade-off of having fully automated system. It is important to notice that DME has attempted modeling of every variable present in the data set, but only Family was found as dependant and thus, reported. Again, non-target variable Flavanoids was found as successfully modeled, which is supported by correlation between this and variable Phenols.

8.1.3 Adult data set

Data set was extracted from publicly available data base in 1994. It contains relatively clean record of citizens. The task is to predict whether a person earns more or less than 50 000$ per year based on different social, educational and marital factors. The only artificially generated attribute is "weighted tallies" which reflect social-demographic characteristics of a person. The data set characteristics are in Table 6 and the report generated is in Figure 36

Data set	Attributes	Task	#Instances	#Attributes	#Web hits
Multivariate	Categorical Integer	Classification	48842	14	464768

Table 6 Adult data set characteristics [40]

The real target variable Earnings was predicted with 0.83 accuracy while the best known 0.8595 (Kohavi, 1996) using NBTree algorithm. Latter work proposed similar accuracy of

[40] https://archive.ics.uci.edu/ml/datasets/Adult

0.8505 using variation of SVM algorithm (Sun, 2003). DME found that SVM was the most successful.

Variables present in the data set are :

1 Age

2 Work class

3 Weighted tallies

4 Education

5 Education years

6 Marital status

7 Occupation

8 Relationship

9 Race

10 Sex

11 Capital gain

12 Capital loss

13 Hours per week

14 Native

15 Earnings

Variable Education years can be predicted with average of 0.00 relative mistake using Linear regression algorithm

---

Value " Divorced" of variable "Marital status" can be predicted with 0.86 accuracy using K- nearest neighbor algorithm

Value " Husband" of variable "Relationship" can be predicted with 0.99 accuracy using Decision tree algorithm

Value " Female" of variable "Sex" can be predicted with 0.83 accuracy using K- nearest neighbor algorithm

Value " <=50K" of variable "Earnings" can be predicted with 0.83 accuracy using Support vector machine algorithm

---

This paragraph reports correlations which are more than three standard deviations higher than mean of all correlations

Significant correlation between continuous variables were not found

Significant correlation between continuous and categorical variables were not found

If Work class is Private, in 0.34 cases Education will be HS-grad

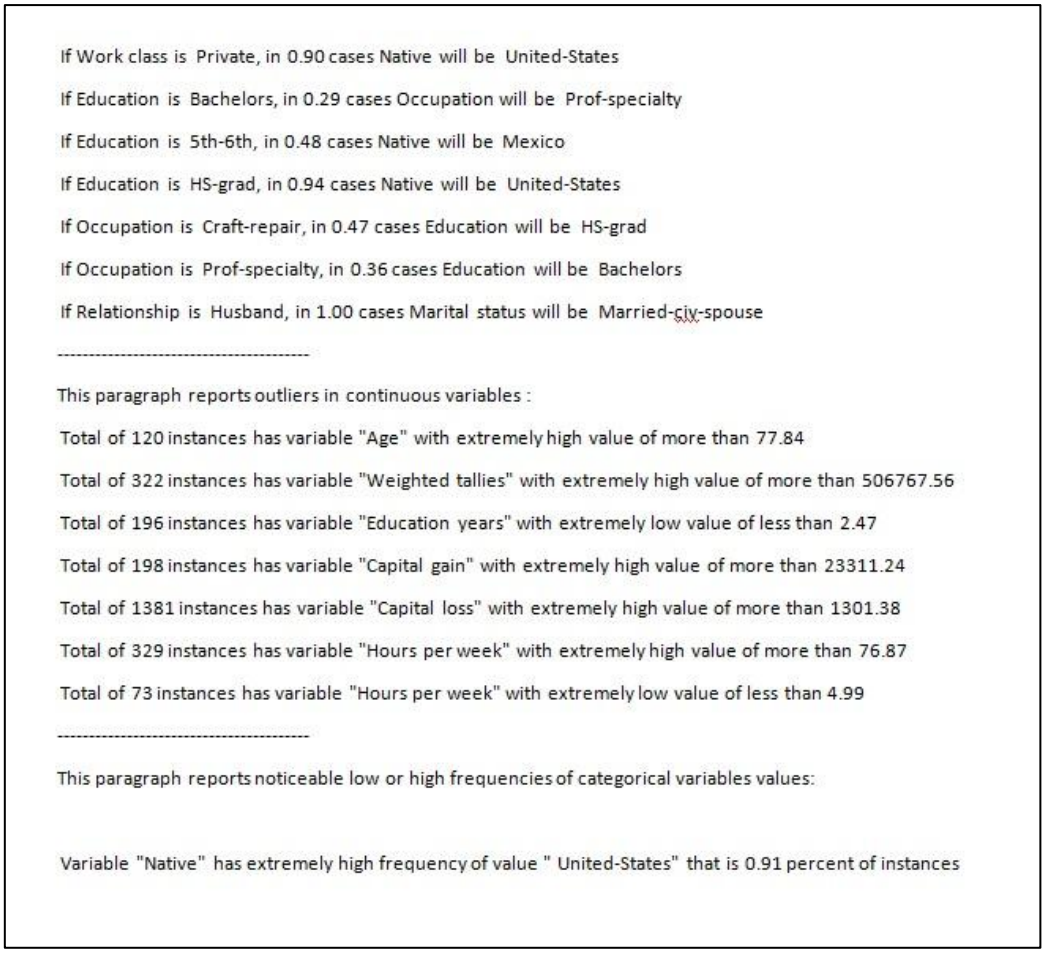


Figure 36 Adult data set report

The data set contains two variables Marital status and Relationship which are obviously strongly correlated so both of them were successfully modeled. Correlations found can be again confirmed as true, for example, it is to be expected that people working as Craft-repair are not highly educated but are mostly High school graduates . DME as well reported some intuitively correct fact such as that there are people with extremely low number of years in education, that is less than 2.47 , same as number of people working more than 76.87 hours per week.

8.1.4 Forest fire data set

The data set includes information about fire occurrences in Portugal. It contains location where it started, time and weather conditions at that time. It also includes four indexes, FFMC, DMC, DC and ISI which are calculated according to Canadian forest Fire Weather Index system (FWI). The goal is to predict the area of the forest which will be burned by fire. The main characteristics of the data set are given in a Table 7.

Data set	Attributes	Task	#Instances	#Attributes	#Web hits
Multivariate	Real	Regression	517	13	183400

Table 7 Forest fire data set characteristics

Variables present in the data set are :

1 X-axis

2 Y-axis

3 month

4 day

5 FFMC

6 DMC

7 DC

8 ISI

9 temp

10 RH

11 wind

12 rain

13 area

Value "3" of variable "X" can be predicted with 0.84 accuracy using K- nearest neighbor algorithm

Value "9" of variable "X" can be predicted with 0.93 accuracy using Decision tree algorithm

Value "aug" of variable "month" can be predicted with 0.98 accuracy using K- nearest neighbor algorithm

Value "jul" of variable "month" can be predicted with 0.98 accuracy using K- nearest neighbor algorithm

Value "mar" of variable "month" can be predicted with 0.99 accuracy using Decision tree algorithm

Value "sep" of variable "month" can be predicted with 0.99 accuracy using Support vector machine algorithm

Value "fri" of variable "day" can be predicted with 0.84 accuracy using K- nearest neighbor algorithm

Value "sat" of variable "day" can be predicted with 0.92 accuracy using Decision tree algorithm

Value "sun" of variable "day" can be predicted with 0.85 accuracy using Decision tree algorithm

Value "tue" of variable "day" can be predicted with 0.91 accuracy using K- nearest neighbor algorithm

---

Variable FFMC can be predicted with average of 0.20 relative mistake using SVR algorithm

Variable area can be predicted with average of 0.12 relative mistake using SVR algorithm

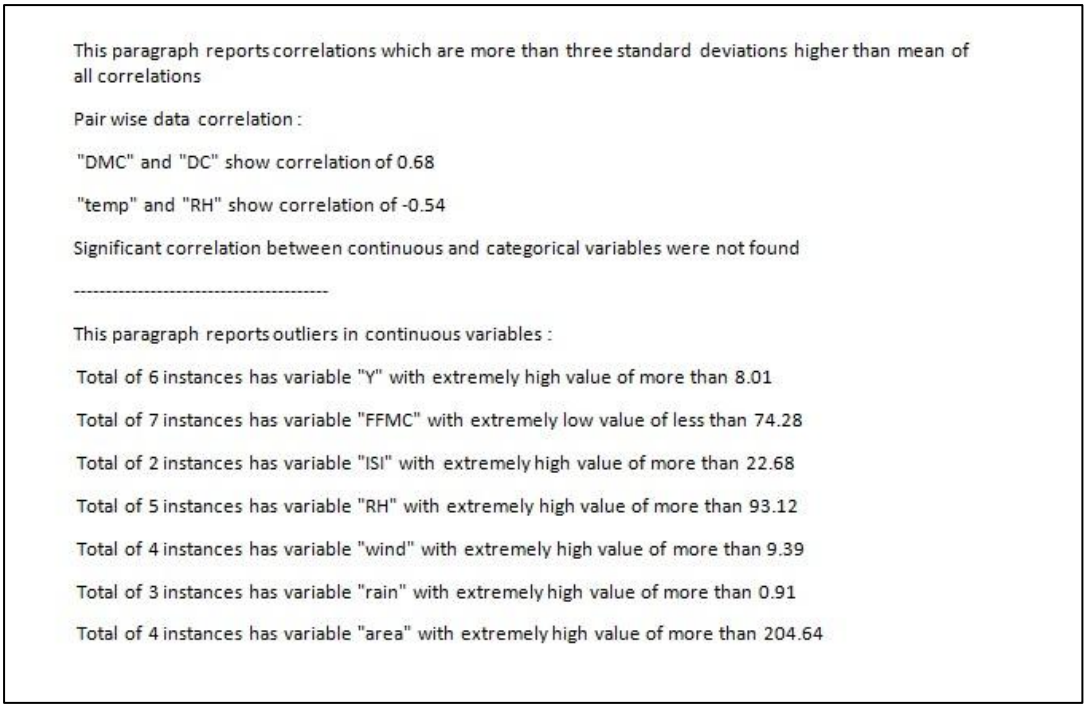


Figure 37 Forest fire data set report

The report which was generated by DME is in Figure 37. Considering that correlations were not found between X-axis and any other variable, we can assume that in these two location, fires are more predictable. The predictions of day and month are unfortunately highly likely result of skewed data, which cannot be prevented. Finally, relative error for area of burned forest is 0.12%.

The highest accuracy reported for this data set is in (Cortez & Morais, 2007) where using support vector machines the authors achieved RMSE of 12.71. Although DME also applies SVM for regression, it shows how important tuning of the parameters is, as well as variable selection to which the authors invested plenty of attention. RMSE reported by DME is 16.71.

8.1.5 Boston Housing data set

The data set is created by (Harrison & Rubinfeld, 1978) and it is considered as difficult regression task since housing values in suburbs of Boston are affected by many other complex factors. The main characteristic are in Table 8. It is still very popular in ML community, but also in social and ecological studies as it displays how for example, prices of houses are affected by percentage of black people in area, or by air pollution.

Data set	Attributes	Task	#Instances	#Attributes	#Web hits
Multivariate	Cat., Real,	Regression	506	14	127021

	Integer				

Table 8 Housing data set characteristics

The values of the variables are mostly given in index form and as such are not very intuitive but by analyzing the variables whose modeling was found possible, we can still confirm that DME delivered expected results.

Variables present in the data set are :

1 Crime rate

2 Residential zone

3 Industry area

4 River side

5 Nitric oxides

6 Room per dwelling

7 Age of homes

8 Distance to work centers

9 Access to highways

10 Tax rate

11 Education

12 Race index

13 Population status index

14 Value of home

Variable Crime rate can be predicted with average of 0.17 relative mistake using SVR algorithm

Variable Nitric oxides can be predicted with average of 0.26 relative mistake using SVR algorithm

Variable Distance to work centers can be predicted with average of 0.28 relative mistake using SVR algorithm

Variable Value of home can be predicted with average of 0.21 relative mistake using SVR algorithm

---

Value "4" of variable "Access to highways" can be predicted with 0.99 accuracy using Decision tree algorithm

Value "5" of variable "Access to highways" can be predicted with 0.97 accuracy using Support vector machine algorithm

Value "6" of variable "Access to highways" can be predicted with 0.98 accuracy using Decision tree algorithm

Value "7" of variable "Access to highways" can be predicted with 1.00 accuracy using Support vector machine algorithm

---

This paragraph reports correlations which are more than three standard deviations higher than mean of all correlations

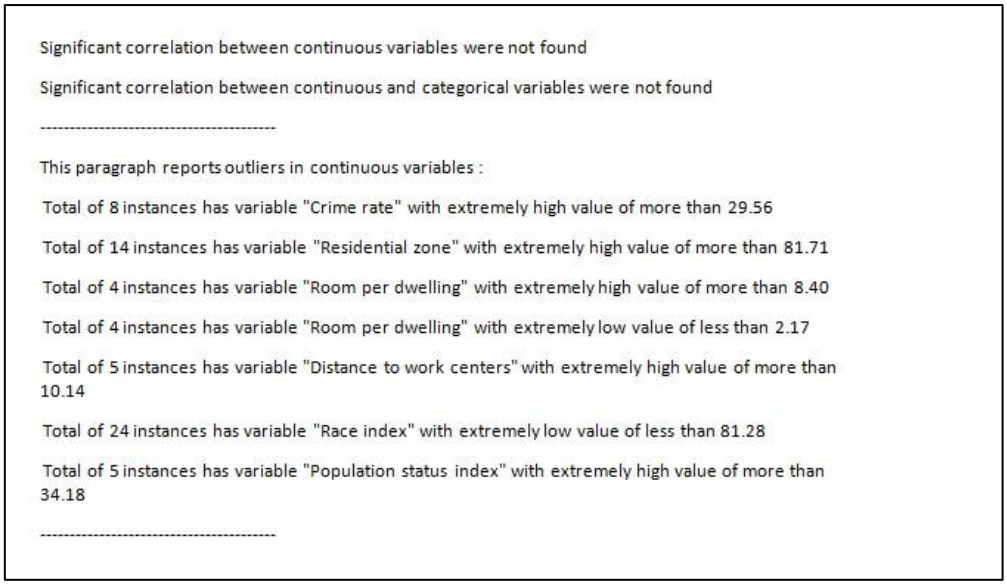


Figure 38 Boston housing data set report

For example having variables such as surface if industrial area and closeness of highways, implies that pollution of air could be predicted. The report also implies that data set is skewed toward upper scale housing, since high number of instances has high population standard index and low Race index.

The best results found are given in (Che, 2013), where the authors used advanced training subset selection. Their best reported RMSE for values of houses is 2.404 while DME achieved 3.32. Although it might seem like a large difference, in reality it is not since the variable in question has mean value 22.56.

8.1.6 Auto mpg data set

The data concerns city-cycle fuel consumption in miles per gallon, to be predicted in terms of 3 multivalued discrete and 5 continuous attributes. The data set is very popular for comparison of regression algorithms but also for clustering or even classification. The main characteristics are given in Table 9.

Data set	Attributes	Task	#Instances	#Attributes	#Web hits
Multivariate	Categorical Real	Regression	398	8	127668

Table 9 Auto mpg data set characteristics

The best found regression is in (Solomatine & Shrestha, 2004) where the AdaBoost algorithm for regression gave RMSE 2.84. which is only slightly better than RMSE of 3.02 achieved by DME using support vectors for regression. Although the models selection is based on MSE,

the report presented to end user states the regression success in terms of relative mistake as it is more intuitive for non-expert end-users. The report for this data set is given in Figure 39.

Variables present in the data set are :

1 mpg

2 cylinders

3 displacement

4 horsepower

5 weight

6 acceleration

7 model year

8 origin

Variable mpg can be predicted with average of 0.19 relative mistake using SVR algorithm

Variable cylinders can be predicted with average of 0.13 relative mistake using SVR algorithm

Variable displacement can be predicted with average of 0.11 relative mistake using SVR algorithm

Variable horsepower can be predicted with average of 0.16 relative mistake using Polynomial curve fitting algorithm

---

Value "2" of variable "origin" can be predicted with 0.94 accuracy using Decision tree algorithm

Value "3" of variable "origin" can be predicted with 0.89 accuracy using Decision tree algorithm

---

This paragraph reports correlations which are more than three standard deviations higher than mean of all correlations

Pair wise data correlation :

"cylinders" and "displacement" show correlation of 0.95

"cylinders" and "weight" show correlation of 0.90

"displacement" and "horsepower" show correlation of 0.90

"displacement" and "weight" show correlation of 0.93

"horsepower" and "weight" show correlation of 0.86

Significant correlation between continuous and categorical variables were not found

---

This paragraph reports outliers in continuous variables :

Total of 4 instances has variable "horsepower" with extremely high value of more than 220.57

Total of 2 instances has variable "acceleration" with extremely high value of more than 23.67

Figure 39 Auto mpg data set report

8.1.7 Conclusions

All of these data sets are publicly available in different machine learning data set repositories and the machine learning community uses them to compare learning algorithms. At the

same time the community maintains the list of the best reported accuracies which can be used to validate our automated system accuracy. Comparison for the classification and regression testing is given in Figure 40 and Figure 41.

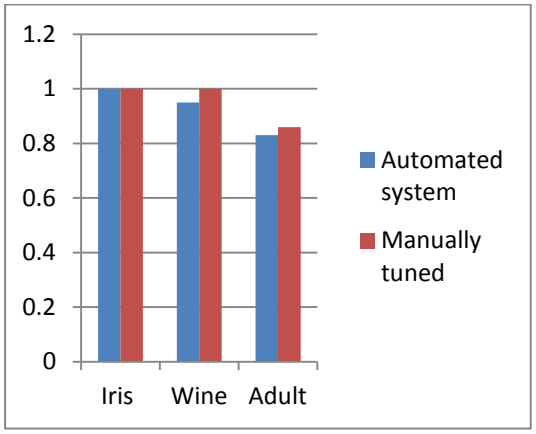


Figure 40 Results for classification accuracy [%]

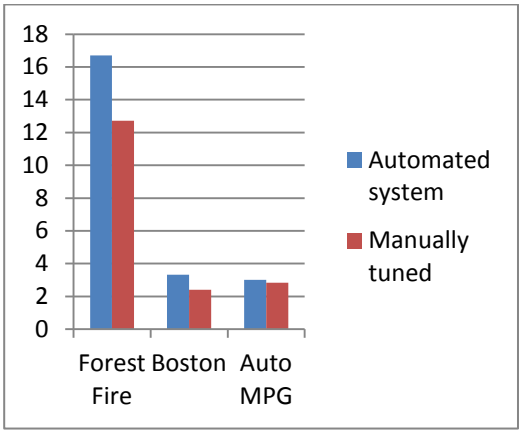


Figure 41 Results of regression testing [RMSE]

It can be noted that the performance of the automated system is equal to or comparable to the best manually tuned algorithms. The representation of relevant information is occasionally cluttered with number of less relevant findings and that might cause problems with high-dimensional data sets. That becomes obvious in these tests where the target variable is known in advance and it might happen in some PLM applications as well. Unfortunately, at this stage, such result delivery still remains the best available solution, since the assumption is that the system is fully automatic, without any input from the user and exists in order to detect non-assumed patterns.

Chapter 9 System testing

9.1 Quality control use case

Quality control use-case refers to hot stamping process and aims at analyzing correlations between production process parameters, geometric and micro structural properties, and cracks revealed within work pieces, thus enabling a better quality and reducing errors and defects and finally improve and adapt production process. In order to automatically detect pieces with defects, 3D scanner is implemented, which generates CAD representation of a manufactured piece. Errors are then detected by comparing the CAD generated and the one of the ideal Master piece.

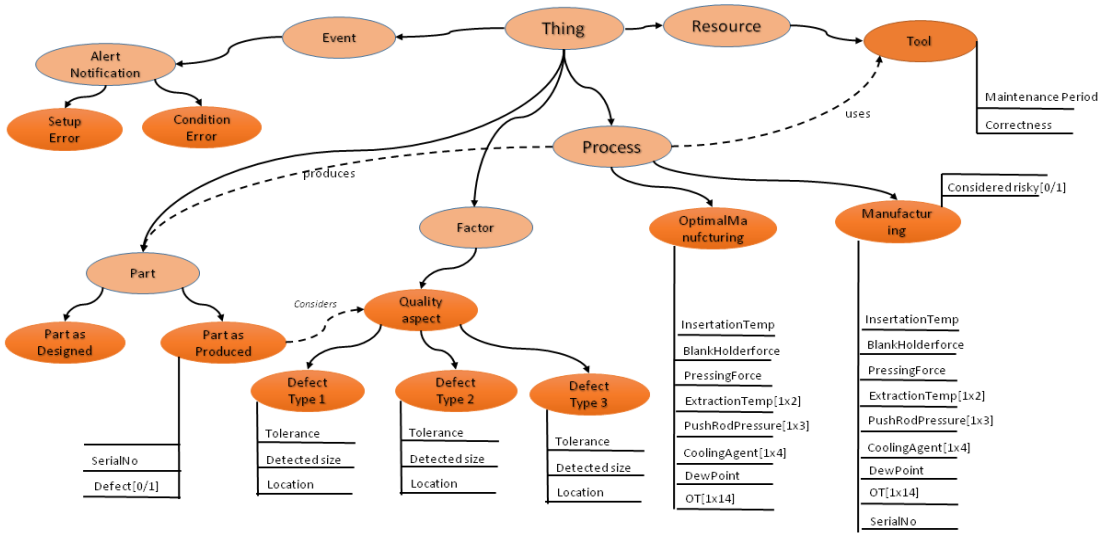


Figure 42 Quality control ontology

Designed ontology is presented in Figure 42. Concepts marked with lighter color are those present in upper ontology, while the darker are designed to model static and dynamic details of the domain.

9.1.1 Use case functional requirements

A scanning process is time-costly and it requires multiple more time than the hot stamping process itself, leading to conclusion that it is not feasible to scan each piece produced. The implemented solution is to monitor different conditions during the process, such as temperature, pressure and time, detect when some of these parameters might be a cause of crack and declare that piece as "risky". The piece in question will be than scanned in order to determine whether there in fact exists a defect. This means that scanning is eliminated as bottle-neck of production procedure, as only a certain subset of pieces will be scanned. The solution relays on assumptions that cause of defects lays in monitored parameters. The assumption was drawn from company workers and their experience in current manual quality control. In order to embed these heuristics into ontological knowledge base, we have interviewed the workers and modeled what they treat as critical parameters values into ontological rules. The process and its parameters in question is illustrated in Figure 43.

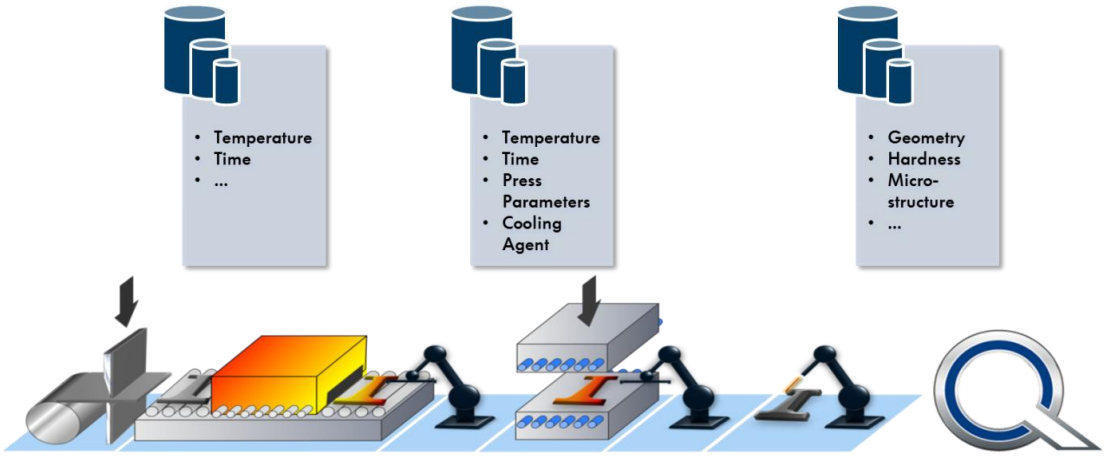


Figure 43 Illustration of process subject to quality control

As the ontology is designed so that it covers in details the entire procedure of quality control, ontology rules are defined to support the scanning results capturing, as well as appropriate actions for when the defect is detected.

9.1.2 Rule groups

Rule base set for quality control defined for this use-case contains three types of rules. The SWRL implementation of all rules are give in the Appendix D of this dissertation.

(1) Production monitoring for purpose of detecting “risky” pieces. These rules are defined so that they compare the current production parameter values and label as Risky those pieces that were produced with parameters outside the predefined tolerance intervals.

(2) Rules that capture scanning process and defect detection. These rules define when the defect is detected, or more precisely, what is the tolerance for dimensions of defects.

(3) Rules for alerting appropriate actors in a case of defect. These are the rules that trigger appropriate mechanisms for notifying responsible personnel.

9.1.3 The data mining module result

In Figure 42, parent-child relations are represented with a full line, while relations between concepts are in dash-line. This means that every dash-line generated at least one data set and some of them generated more, for every child of two concepts in questions. As assumed number of data sets gave no useful result. For example merging attributes of Part as Produced and OptimalManufacturing will give no results since all the attributes except Defect are constant for all the instances. Similar situation is for data set combining Quality Aspects and Part as Produced. Still, since the system is fully automatic and self-initiated, these results will stay unnoticed and will not become additional burden for workers. On the other hand, the data set created by merging Part as Produced and Manufacturing, gave some interesting results. As usual, they are delivered in form of report presented in Figure 44.

The report starts with the most relevant finding which is that presence of defect can be predicted with 81% accuracy. The worker in charge gains several useful conclusions from this information. First is that it is justified to assume that defects are caused by manufacturing parameters to a quite high degree, but not fully. It supports the solution where only "risky" pieces are scanned, but scanning cannot be avoided. End-user also gets an information that the optimal algorithm is decision tree, in a case that he needs to explore further and find out which variables are the most relevant. Following, correlations reported are the expected ones. Next information is high correlation between Defect and temperature in last zone of the oven, which is useful as discovery that the extraction temperate is the most relevant for quality of manufacturing. Finally, outliers are reported, as a useful suggestions for maintenance of manufacturing tools.

Important detail to notice is that based on user feedback, the system was updated so that it doesn't report modeling possibility in cases where correlation between two variables is considerably higher than modeling accuracy. This led to omitting of report on modeling for Pushrod pressure and CoolingAgent.

```
---------------------------------------------------------------
Variables present in the data set are :
 1  SerialNo
 2  Defect
 3  InsertationTemp
 4  BlankholderForce
 5  PressingForce
 6  ExtractionTemp1
 7  ExtractionTemp2
 8  PPP1 Pyrometer Platine 1
 9  PPP2 Pyrometer Platine 2
 10  PuchRodPressure1
 11  PuchRodPressure2
 12  PuchRodPressure3
 13  CoolingAgent1
 14  CoolingAgent2
 15  CoolingAgent3
 16  CoolingAgent4
 17  DewPoint
 18  OT01
 19  OT02
 20  OT03
 21  OT04
 22  OT05
 23  OT06
 24  OT07
 25  OT08
 26  OT09
 27  OT10
 28  OT11
 29  OT12
 30  OT13
 31  OT14
---------------------------------------------------------------
```

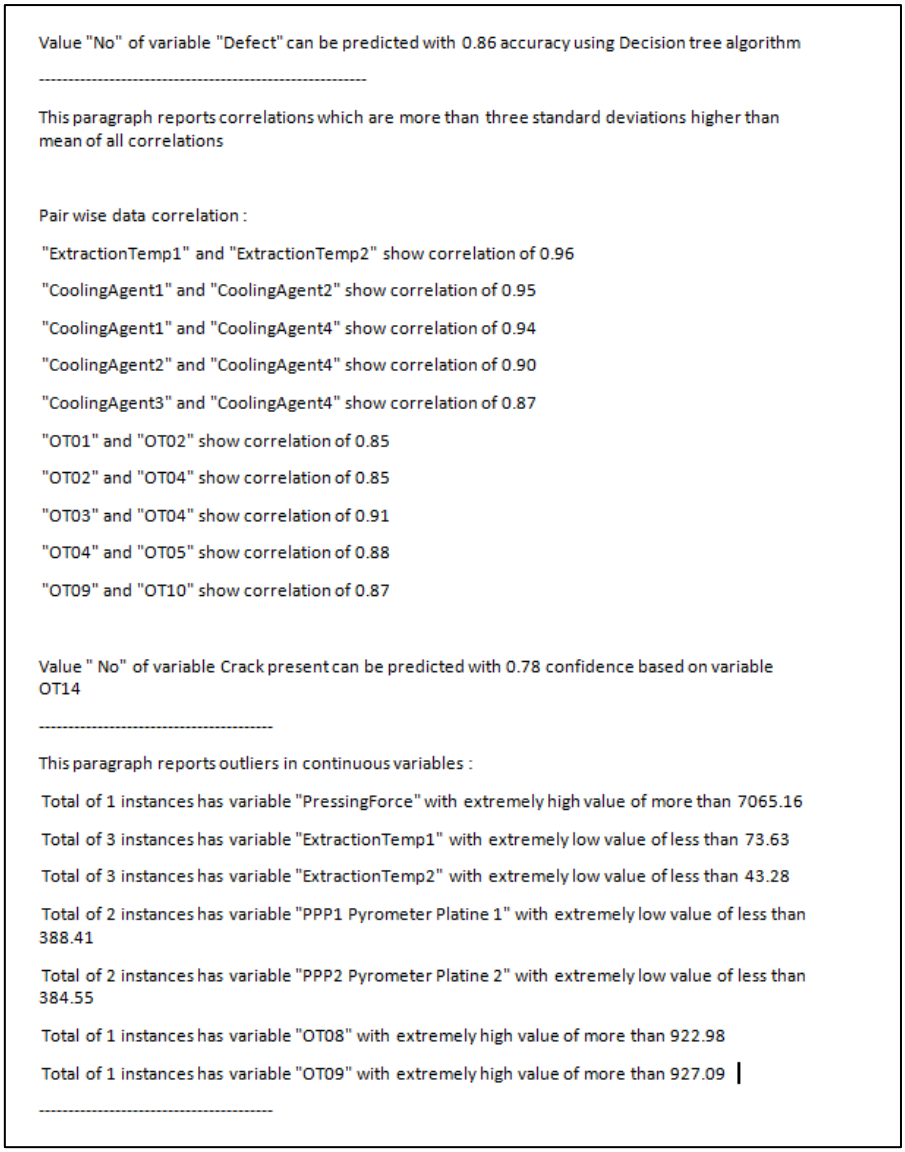


Figure 44 Quality control report

Considering that in this case decision tree algorithm showed to be the most accurate in predicting presence of "Defect", additional visualization can be delivered, as shown in Figure 45. Consequently, cut-off values of critical parameters can be determined and, if validated, they can be defined as additional rules for ontology reasoning exploitation.

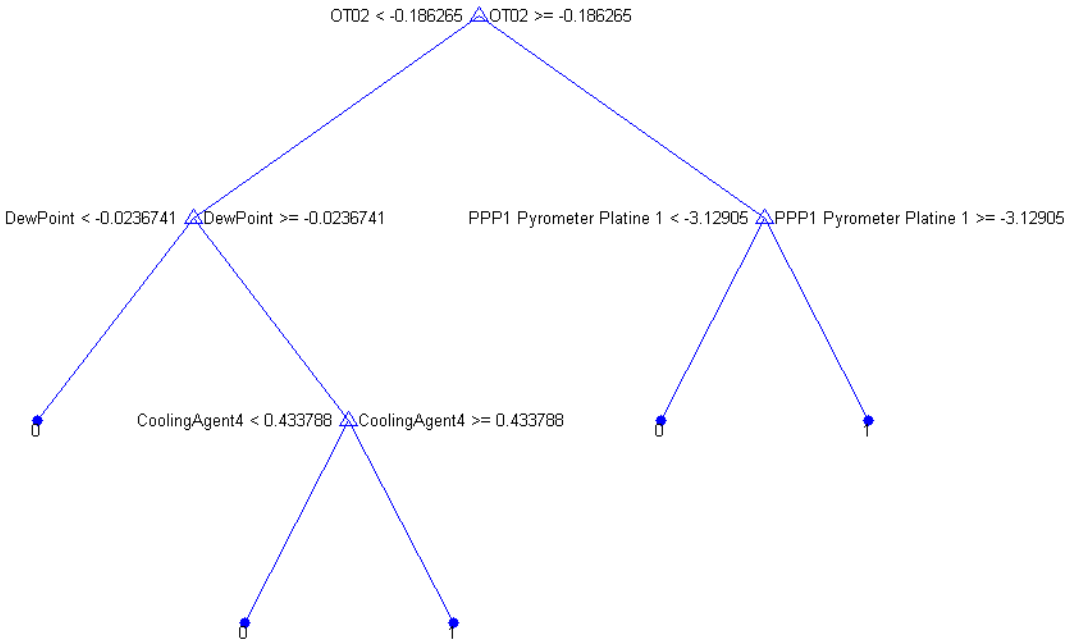


Figure 45 Decision tree for prediction of "Defect" presence

9.1.4 Conclusions

Modeling manufacturing process, together with quality control procedure, into one knowledge base and by exploiting on reasoning capabilities of ontology, leads to system is which light and clear to grasp. In the same time, it helps enabling fast and efficient updates without conflicts or contradictions. Following, by exploiting this knowledge base as ground layer for data mining, we gain even more detailed and observant control of manufacturing process, together with notion of patterns that might be an added value.

9.2 Predictive maintenance use case

Predictive maintenance use case is focused on maintenance of diesel engines of large construction machines. Estimating schedule of servicing activities for such engines is a complex optimization problem. On one hand, if machines are serviced too often, expenses of the consumable resources will be unnecessarily high and the availability of the machines will be shortened. On the other hand, if the engine is not maintained properly and the breakdown occurs, the expenses will be even higher. To be able to estimate this schedule, number of factors from different perspectives needs to be considered.

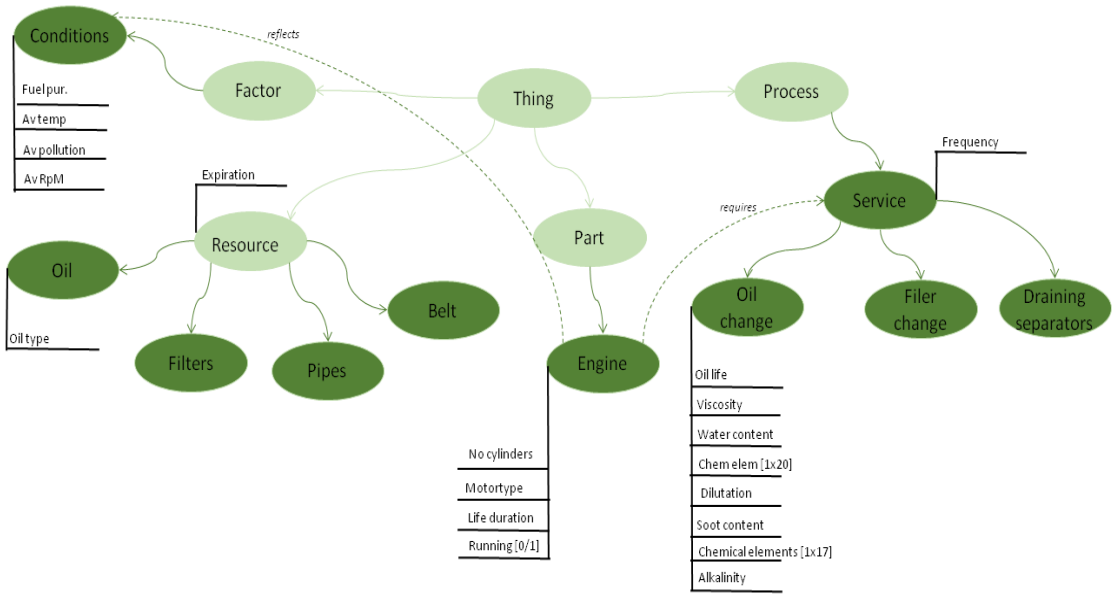


Figure 46 Predictive maintenance ontology

Designed ontology is presented in Figure 46. Concepts marked in lighter color are those present in upper ontology, while the darker are domain specific.

9.2.1 Use case functional requirements

Oil change is considered to be the main factor in prolonging the engines life as the oil properties will reflect conditions of most of the other factors. Air filters purity, engine running conditions, diesel fuel quality etc. are all affecting the oil chemical composition and thus performance. Through extensive testing and analysis, thresholds for content of undesirable chemical compounds in oil have been determined. For example, most engine manufacturers recommend that solids shouldn't excide more than 5%. Still, performing frequent chemical analysis of the oil is time consuming and not feasible.

Based on experience and common knowledge as well as manufacturers' recommendations, the factors that affect the oil life have been modeled in the ontology and the rules are designed to capture variability of oil draining frequency.

9.2.2 Rule groups

Rule base set for predictive maintenance defined for this use-case contains three types of rules. The SWRL implementation of all rules are given in the Appendix E of this dissertation.

(1) Rules for servicing based on resource manufacturers recommendations. These rules are used to capture standardized periods of consumable expiration, such as oil change or filter change based on the type of resource and environment
(2) Rules for assuming a malfunction of an engine base on known critical values of chemical elements content in the oil. This knowledge is based on tests and analysis performed by engine manufacturers and independent groups.
(3) Rules for adjusting the maintenance periods based on environmental conditions. This knowledge is obtained through experiments and analysis of research groups in industry and academia.

9.2.3 The data mining module result

Presented in Figure 47 is a report for data set created by merging attributes of Engine and Service. Classification attempts for categorical attributes gave two models. First one is for type of motor prediction, which is consequence of having number of cylinders as variable and very skewed sampling in the data set. It doesn't bring a new knowledge about the predictive maintenance. The second, on the other hand, gives a high accuracy of motor failure prediction. Accuracy of 91% leads to conclusion that motor failure indeed highly depends on oil condition. Interesting detail is that the best performing algorithm is K-nearest neighbor, which indicates high non-linearity and certain complexity in the pattern.

Correlation between chemical elements levels are reported for elements which are consequences of the same causes. Zinc and phosphor co-occur as they are both present in well-know ZDDP additives. The report also gives a long list of outliers which is due to the fact that the ontology is designed for predictive maintenance, thus high percentage of engines has extreme conditions.

Variables present in the data set are :

1 No cylinders
2 Motortype
3 Life duration
4 Running
5 Oil life
6 Viscosity
7 Water content
8 Soot content
9 Ba
10 Ca
11 Mg
12 P
13 Zn
14 B
15 K
16 Si
17 Na
18 Al
19 Cr
20 Cu
21 Fe
22 Pb
23 Ni
24 Ag
25 Sn
26 Ti
27 Dilutation
28 Alkalinity

Value "D934 A7" of variable "Motortype" can be predicted with 0.93 accuracy using Decision tree algorithm

Value "No" of variable "Running" can be predicted with 0.91 accuracy using K-nearest neighbor algorithm

---

Variable Al can be predicted with average of 0.05 relative mistake using SVR algorithm

Variable Dilutation can be predicted with average of 0.06 relative mistake using SVR algorithm

---

This paragraph reports correlations which are more than three standard deviations higher than mean of all correlations

Pair wise data correlation :

"P" and "Zn" show correlation of 0.94

"Al" and "Fe" show correlation of 0.96

This paragraph reports outliers in continuous variables :

Total of 11 instances has variable "Life duration" with extremely high value of more than 4021.55

Total of 3 instances has variable "Oil life" with extremely high value of more than 2155.94

Total of 1 instances has variable "Viscosity" with extremely low value of less than 9.81

Total of 1 instances has variable "Water content" with extremely high value of more than 1.54

Total of 2 instances has variable "Ca" with extremely low value of less than 1159.23

Total of 2 instances has variable "Mg" with extremely high value of more than 32.23

Total of 1 instances has variable "P" with extremely low value of less than 117.16

Total of 2 instances has variable "K" with extremely high value of more than 108.44

Total of 2 instances has variable "Al" with extremely high value of more than 104.01

Total of 4 instances has variable "Cr" with extremely high value of more than 7.35

Total of 8 instances has variable "Cu" with extremely high value of more than 261.74

Total of 2 instances has variable "Fe" with extremely high value of more than 295.69

Total of 5 instances has variable "Sn" with extremely high value of more than 8.69

This paragraph reports noticeable low or high frequencies of categorical variables values:

Variable "Motortype" has extremely high frequency of value "D934 A7" that is 0.15 percent of instances

Figure 47 Predictive maintenance report I

Visualization example in Figure 48 shows the patter of Iron and Copper levels influence to engine failure. These two variables were chosen as the two with the highest correlation to engine failure. As assumed, the decision bound is highly non-linear without clear cut-off values, which indicates that engine failure must be a consequence of more complex combination of conditions. Still, it is definite that the increased levels of any one of these elements are indicators of engine failure.

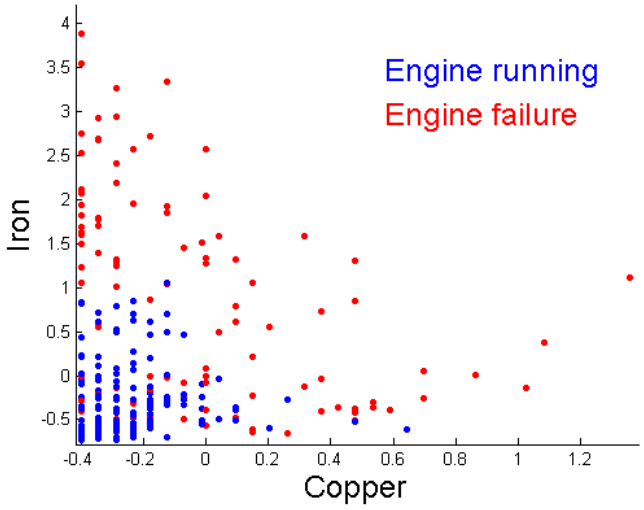


Figure 48 Example I of engine failure pattern

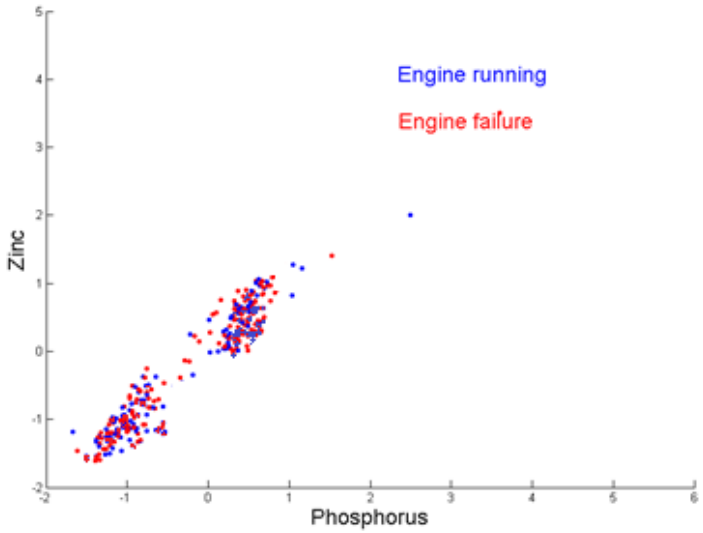


Figure 49 Example II of engine failure pattern

Visualized example in Figure 49 shows strong correlation between levels of phosphorus and zinc indicating that these two elements are elevated due to same cause. The cause itself is actually that both elements are present in oil additives that are used for high-power engines. There is also a slight correlation between levels of these two elements and engine failure, indicating that low levels of these elements are connected to engine failure.

Variables present in the data set are :

1 No cylinders
2 Motortype
3 Life duration
4 Running
5 Fuel purity
6 Average temperature
7 Average pollution
8 Average mileage

Value "No" of variable "Running" can be predicted with 0.76 accuracy using Support vector machine

---

This paragraph reports correlations which are more than three standard deviations higher than mean of all correlations

Pair wise data correlation :

"Average mileage" and "Average pollution" show correlation of 0.84

This paragraph reports outliers in continuous variables :

Total of 11 instances has variable "Life duration" with extremely high value of more than 4021.55

Total of 3 instances has variable "Average temperature" with extremely high value of more than 31.78

Total of 1 instances has variable "Average temperature" with extremely low value of less than 1.34

Figure 50 Predictive maintenance report II

Second report was generated based on relation between Engine and Conditions. It is known that environmental factors influence that oil performance. For example, in high temperature, viscosity of the oil is decreased and vice versa. Even with such limited number of variables, failure of the engine can still be modeled. Outliers here, again point out that the engines present in the data set have been working in extreme conditions.

9.2.4 Conclusions

Modeling factors that are relevant for oil conditions and thus engine functioning brings clear picture of complexity of the problem. Rule reasoning will result in adaptive scheduling based on empiric knowledge and real-life data gathered in run time. Data mining results bring additional benefits in a form of confirmation of hypothesis on one side and refinement of critical oil conditions.

Important conclusion that can be drawn from this use case is that our system has one important limitation. It is justified to assume that in this case, combining more than two concepts into data set would be beneficial. If the same variable, that is whether the engine is in the running condition can be modeled using two disjoint sets of variables, than it would be safe to assume that accuracy of classification would be increased by combining all three concepts' attributes. Although this step wouldn't bring nothing new in a conceptual level, there is a number of technical challenges to consider and it will be done in future work.

Chapter 10 Conclusion

Contributions to this research started with recognizing a need in the PLM community for an efficient and easily comprehensive method for managing knowledge of the domain in question. It adopts ontology as the most recent trend in semantic technologies and accordingly, proposes a fresh method for the simplification and acceleration of the ontology design process. Further on, ontology rule inference has been used mostly to describe inheritance and static results for rule chaining. This Dissertation brings a different perspective to ontological rule exploitation that is to enrich the spectrum of ontology functionalities. Finally, the question of an efficient extraction of patterns from data in the PLM is addressed. By combining the benefits of ontology and by introducing compromises for pattern precision, a fully automatic data modeling module is designed.

10.1 Achieved results

Managing knowledge and data related to a product can be considered a complex issue, due to the high number of actors, data sources and processes present in one of, or all, stages of a product's life cycle. This creates a need to gather and structure large and diverse amounts of information into one unified model, which can be accessed, updated and reused by all relevant actors. Ontology emerged as a convenient solution due to its strong attributes such as providing a comprehendible overview of the domain, interoperability with existing structures, human understandable structure and easy updates. A considerable amount of work on ontologies, for specific parts of the PLM domain, has been carried out in academic, as well as, industrial circles. In order to bring a generic template of ontology for PLM, and to provide an efficient method for ontology design, a user story mapping procedure coming from the software design field was exploited. Applying this procedure in order to gather a domain's existing knowledge, creates the first contribution to this Dissertation. Following the same steps, as when gathering user's requirements for software functionalities, it is possible to gather perspectives, experience and expectations from all relevant actors. This information is then transformed, in a straightforward manner, into a list of concepts and relations. By performing the generalization of concepts, the upper ontology for the PLM domain was created and as such, it brings added value to this research. Upper ontology can be con-

sidered as a template for future ontological design, where semantic experts can use it as a reference model, instead of starting the process from the beginning every time.

Ontological reasoning is a mechanism that is used to model the inheritance of properties between concepts, such as, that all instances of sub-class have the same properties as super-class. It is further extended to reasoning by using the first order of rule chaining, where new relations are defined by the existence of assumptions. For example, hasChild and hasBrother will result in the creation of hasUncle relation, since in reality, uncle is a relation defined as a consequence of existing relations. Finally, ontology reasoning can result in an attribute value assignment. All of these tools, come as immeasurably useful in the PLM domain, where complex processes affect a domain state in multiple manners. Yet, besides defining rules to reflect the state of the domain, rules are exploited to enable diverse additional functionalities of an ontology in this Dissertation.

To present the capabilities of this strategy, we tested the approach on three real life use cases. In the first case, we addressed the issue of employing industrial standards during the product design stage, where consulting an unstructured text of a standard is time consuming and unreliable. By designing an ontology that covers the domain of the standard, contained information become structured and thus more convenient to brows and query. Added value is gained by defining ontology rules so that when integrated with a CAD system, ontology reasoning can provide assistance to the designer and worn him when a standard's regulations are being violated. The second use case addresses a life cycle cost optimization problem. Ontological rules were used to insure that the product is still in accordance with a customer's request at minimal cost. Furthermore, ontology rules were used for enabling a complex inheritance of properties from part to a product. Finally, product information from the operational phase were controlled and exploited using ontology rules. In the third use case, rules were used to automatically define and maintain validity of knowledge, in the knowledge base, based on a number of external factors.

Through use cases, it has been illustrated that ontology rules can adopt a number of functionalities that will result in a contribution to knowledge structure as well as ontology adaptability and thus usability.

As the third contribution to this Dissertation, a data mining module was designed that, by relaying on the ontology and sub-optimality assumption, performs a fully automatic data analysis. This module was designed to address the gap which exists in PLM domain data exploitation, where human experts in data analysis rarely have the overview of the entire domain and all relevant factors. As a consequence, a number of cause-effect relations remains undetected, especially between concepts between different life stages. Having ontology as a model for the entire domain opens the door to the automatic extraction of data sets which potentially contain relevant patterns. The data mining module will perform analysis in a fully

automatic manner, but without a guarantee that patterns will be optimally precise. We argue that in the PLM domain, detecting a relation with some uncertainty is more valuable than not detecting the relation at all. The key benefit is the result of automation and human readable output, people who are non-experts in data mining can use the system.

It might seem as redundant to build ontology from unstructured data, and then to once again extract tabular data for data mining purposes. An important aspect is that ontology is not designed only for data analysis purposes. Ontology in PLM is designed with a number of advantages that come with semantic technologies and unified domain models. Data mining is designed as an added value to existing ontology and an introduction to an entire new direction of ontological exploitation.

This work included state of the art methods and algorithms from three very different research fields that is PLM, ontology and data mining. As such, it displays that added value is created by combining advantages of one discipline to address gaps in another.

10.2 Future development

To our knowledge, this is the first system with this exact symbiotic structure, and as such it is only the first prototype. A number of improvements can be imagined and tested. First of all, the data mining module can be improved with more refined algorithms and heuristics so that it provides more accurate models and conclusions. The obstacle to applying more complex algorithms is that the execution time might increase dramatically. This would be a consequence of the higher complexity of training procedures as well as a higher number of parameters to select from. Another question for future research is how to handle big data. Sampling instances can remedy the problem to a certain point, after which it will become infeasible to expect results. The answer might lie in applying fast online learning algorithms and stopping the training when the trend starts to be close to constant.

One of the downsides that could benefit from further work is already described in section 9.2.4. Current system extracts attributes of two concepts to create data set. This could obviously be improved by extending a number of concepts to three or even more. The problem is that the number of data sets would grow exponentially and most of them wouldn't generate useful results. For this reason, additional criteria would have to be proposed, based on which only some data sets would be analyzed.

Finally, a simple and easy-to-design improvement would be to give some control to the end user, in a sense that he/she might know in advance what the target variable, which they want to model, is. Having this information would shorten execution time as well as eliminate all irrelevant results from output reports. This could be done by simple user interface with

multiple choice questions. Yet, as such, this improvement could be part of work on developing a commercial tool, of this type, and is not vital for the purpose of this Dissertation

References

Abburu, S. (2012). A Survey on Ontology Reasoners and Comparison. *International Journal of Computer Applications*, *57*(17), 33–39. Retrieved from http://www.ijcaonline.org/archives/volume57/number17/9208-3748

Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., … Cudré-Mauroux, P. (Eds.). (2007). *The Semantic Web* (Vol. 4825). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-540-76298-0

Altman, N. S. (1992). An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician*, *46*(3), 175–185. doi:10.1080/00031305.1992.10475879

Arlinghaus, S. (1994). *Practical handbook of curve fitting*. Retrieved from http://books.google.ch/books?hl=en&lr=&id=nuxxw4AeY_UC&oi=fnd&pg=PA1&dq=Arlinghaus,+Sandra+L.+1994.+(ed.)+"Practical+Handbook+of+Curve+Fittin&ots=taVpIYmvGz&sig=GjTH53vxZQd86r8kCT8JDa9Pj18

Asghar, S., & Iqbal, K. (2009). Automated Data Mining Techniques: A Critical Literature Review. In *2009 International Conference on Information Management and Engineering* (pp. 75–79). IEEE. doi:10.1109/ICIME.2009.98

Basak, D., Pal, S., & Patranabis, D. (2007). Support vector regression. *Neural Information Processing- …*. Retrieved from http://www.researchgate.net/publication/228537532_Support_vector_regression/file/60b7d51c2a0d86d3d9.pdf

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*. Retrieved from http://www.inf.ufsc.br/~kfasolin/Semantic Web/Perguntas aula 1.doc

Berry, M., & Linoff, G. (1999). Mastering Data Mining: The Art and Science of Customer Relationship Management. Retrieved from http://dl.acm.org/citation.cfm?id=555358

Bishop, C. (2006). *Pattern recognition and machine learning*. Retrieved from http://soic.iupui.edu/syllabi/semesters/4142/INFO_B529_Liu_s.pdf

Bohn, R. (1998). Measuring and managing technological knowledge. *The Economic Impact of Knowledge, Butterworth- …*. Retrieved from http://books.google.ch/books?hl=en&lr=&id=xJbS5-IP2lgC&oi=fnd&pg=PA295&dq=+"MEASURING+AND+MANAGING+TECHNOLOGICAL+KNOWLEDGE"&ots=vxdbAyKbEn&sig=2mn-jBqz_Y_9lPUmCUNXF2A0DNg

Boley, H. (2006). *The RuleML family of web rule languages*. Retrieved from http://link.springer.com/chapter/10.1007/11853107_1

Breiman, L., Friedman, J., Stone, C., & Olshen, R. (1984). Classification and regression trees. Retrieved from http://scholar.google.ch/scholar?cluster=3017989415557051150&hl=en&oi=scholarr&sa=X&ei=1wS9VLnEEoTWavD0gqgG&ved=0CBsQgAMoADAA#0

Burges, C. J. C. (1998). A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, *2*(2), 121–167. doi:10.1023/A:1009715923555

Campos, M. M., Stengard, P. J., & Milenova, B. L. (2005). Data-Centric Automated Data Mining. In *Fourth International Conference on Machine Learning and Applications (ICMLA'05)* (pp. 97–104). IEEE. doi:10.1109/ICMLA.2005.18

Casanovas, P., Casellas, N., Tempich, C., Vrandečić, D., & Benjamins, R. (2007). OPJK and DILIGENT: ontology modeling in a distributed environment. *Artificial Intelligence and Law*, *15*(2), 171–186. doi:10.1007/s10506-007-9036-2

Che, J. (2013). Support vector regression based on optimal training subset and adaptive particle swarm optimization algorithm. *Applied Soft Computing*, *13*(8), 3473–3481. doi:10.1016/j.asoc.2013.04.003

Collinson, S. (2006). Inertia in Japanese Organizations: Knowledge Management Routines and Failure to Innovate. *Organization Studies*, *27*(9), 1359–1387. doi:10.1177/0170840606067248

Cortez, P., & Morais, A. de J. R. (2007, December 1). A data mining approach to predict forest fires using meteorological data. Associação Portuguesa para a Inteligência Artificial (APPIA). Retrieved from http://repositorium.sdum.uminho.pt/handle/1822/8039

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27. doi:10.1109/TIT.1967.1053964

d'Amato, C., Fanizzi, N., & Esposito, F. (2008). Non-parametric Statistical Learning Methods for Inductive Classifiers in Semantic Knowledge Bases. In *2008 IEEE International Conference on Semantic Computing* (pp. 291–298). IEEE. doi:10.1109/ICSC.2008.28

Dalwadi, N., Nagar, B., & Makwana, A. (2012). Semantic Web And Comparative Analysis of Inference Engines. *Int. J. of Computer …*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.233.412&rep=rep1&type=pdf

Davenport, T., De Long, D., & Beers, M. (1998). Successful knowledge management projects. *SLOAN MANAGEMENT REVIEW*, *39*(2), 43–+. Retrieved from http://apps.webofknowledge.com/full_record.do?product=WOS&search_mode=Refine&qid=16&SID=S2c4rW27R1FHKohyLmA&page=1&doc=2

Davenport, T. H., & Prusak, L. (1998). *Working Knowledge: How Organizations Manage what They Know, Part 247* (p. 199). Harvard Business Press. Retrieved from http://books.google.com/books?hl=en&lr=&id=-4-7vmCVG5cC&pgis=1

Davies, D. L., & Bouldin, D. W. (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-1*(2), 224–227. doi:10.1109/TPAMI.1979.4766909

De Bie, T. (2011). An information theoretic framework for data mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11* (p. 564). New York, New York, USA: ACM Press. doi:10.1145/2020408.2020497

Do, N., Bae, S., & Park, C. (2015). Interactive analysis of product development experiments using On-line Analytical Mining. *Computers in Industry*, *66*, 52–62. doi:10.1016/j.compind.2014.09.003

Drucker, P. F., & Drucker, P. F. (1994). *Post-capitalist Society* (p. 204). Routledge. Retrieved from http://books.google.com/books?hl=en&lr=&id=IYVBmM5z69cC&pgis=1

Faloutsos, C., & Megalooikonomou, V. (2007). On data mining, compression, and Kolmogorov complexity. *Data Mining and Knowledge Discovery*, *15*(1), 3–20. doi:10.1007/s10618-006-0057-3

Fanizzi, N., d'Amato, C., & Esposito, F. (2012). Induction of robust classifiers for web ontologies through kernel machines. *Web Semantics: Science, Services and Agents on the World Wide Web*, *11*, 1–13. doi:10.1016/j.websem.2011.11.003

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*. Retrieved from http://www.aaai.org/ojs/index.php/aimagazine/article/viewArticle/1230

Fernández-López, M., Gómez-Pérez, A., & Juristo, N. (1997, March 1). METHONTOLOGY: From Ontological Art Towards Ontological Engineering. Facultad de Informática (UPM). Retrieved from http://oa.upm.es/5484/1/METHONTOLOGY_.pdf

Gordon, W. J. (1971). Blending-Function Methods of Bivariate and Multivariate Interpolation and Approximation. *SIAM Journal on Numerical Analysis*, *8*(1), 158–177. doi:10.1137/0708019

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, *5*(2), 199–220. doi:10.1006/knac.1993.1008

Gunasekaran, A. (1999). Agile manufacturing: A framework for research and development. *International Journal of Production Economics*, *62*(1-2), 87–105. doi:10.1016/S0925-5273(98)00222-9

Guo, Y., Pan, Z., & Heflin, J. (2005). LUBM: A benchmark for OWL knowledge base systems. *Web Semantics: Science, Services and Agents on the World Wide Web*, *3*(2-3), 158–182. doi:10.1016/j.websem.2005.06.005

Harrison, D., & Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, *5*(1), 81–102. doi:10.1016/0095-0696(78)90006-2

Hass, K. (2007). The blending of traditional and agile project management. *PM World Today*. Retrieved from http://mx1.chelsoftusa.com/uploads/2/8/3/8/2838312/agile_well_explained.pdf

Hoerl, A. E., & Kennard, R. W. (2012). Ridge Regression: Biased Estimation for Nonorthogonal Problems. Retrieved from http://amstat.tandfonline.com/doi/abs/10.1080/00401706.1970.10488634#.VL5qqCuvCUk

Jun, H.-B., Kiritsis, D., & Xirouchakis, P. (2007). Research issues on closed-loop PLM. *Computers in Industry*, *58*(8-9), 855–868. doi:10.1016/j.compind.2007.04.001

Kiritsis, D. (2011). Closed-loop PLM for intelligent products in the era of the Internet of things. *Computer-Aided Design*, *43*(5), 479–501. doi:10.1016/j.cad.2010.03.002

Kiritsis, D. (2013). Semantic technologies for engineering asset life cycle management. *International Journal of Production Research*, *51*(23-24), 7345–7371. doi:10.1080/00207543.2012.761364

Kiritsis, D., Bufardi, A., & Xirouchakis, P. (2003). Research issues on product lifecycle management and information tracking using smart embedded systems. *Advanced Engineering Informatics*, *17*(3-4), 189–202. doi:10.1016/j.aei.2004.09.005

Klein, D. A. (1998). *The Strategic Management of Intellectual Capital* (p. 246). Routledge. Retrieved from http://books.google.com/books?hl=en&lr=&id=eTBdDsOc9FsC&pgis=1

Kohavi, R. (1996). Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. *KDD*. Retrieved from http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/lecturas-clasificacion/NBTree.pdf

Köksal, G., Batmaz, İ., & Testik, M. C. (2011). A review of data mining applications for quality improvement in manufacturing industry. *Expert Systems with Applications*, *38*(10), 13448–13467. doi:10.1016/j.eswa.2011.04.063

Lachmayer, R., Mozgova, I., Sauthoff, B., & Gottwald, P. (2014). Evolutionary Approach for an Optimized Analysis of Product Life Cycle Data. *Procedia Technology*, *15*, 359–368. doi:10.1016/j.protcy.2014.09.090

Liao, S.-H., Chen, J.-L., & Hsu, T.-Y. (2009). Ontology-based data mining approach implemented for sport marketing. *Expert Systems with Applications*, *36*(8), 11045–11056. doi:10.1016/j.eswa.2009.02.087

Lloyd, J. (1987). *Foundations of Logic Programmin*. Retrieved from http://cgi.di.uoa.gr/~prondo/SEMANTICS/Lloyd.pdf

Matsokis, A., & Kiritsis, D. (2010). An ontology-based approach for Product Lifecycle Management. *Computers in Industry*, *61*(8), 787–797. doi:10.1016/j.compind.2010.05.007

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, *405*(2), 442–451. doi:10.1016/0005-2795(75)90109-9

Mayer, R. J., Menzel, C. P., Painter, M. K., deWitte, P. S., Blinn, T., & Perakath, B. (1995). Information Integration for Concurrent Engineering (IICE) IDEF3 Process Description Capture Method Report. Retrieved from http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA530528

McKenzie-Veal, D. (2010). Implementing Ontology-based Information Sharing in Product Lifecycle Management. *65th Midyear Meeting …*. Retrieved from http://edgd.asee.org/conferences/proceedings/65th Midyear/McKenzie_Veal_Hartman_Ontology_based_Information.pdf

McKinley, S., & Levine, M. (1998). Cubic spline interpolation. *College of the Redwoods*. Retrieved from http://online.redwoods.edu/instruct/darnold/LAPROJ/Fall98/SkyMeg/Proj.PDF

Milicic, A., Perdikakis, A., El Kadiri, S., & Kiritsis, D. (2013). PLM Ontology Exploitation through Inference and Statistical Analysis: Case Study for LCC. In *Manufacturing Modelling, Management, and Control* (Vol. 7, pp. 1004–1008). Retrieved from http://www.ifac-papersonline.net/Detailed/60149.html

Milicic, A., Perdikakis, A., & Kadiri, S. El. (2012). Towards the definition of domain concepts and knowledge through the application of the user story mapping method. *Product Lifecycle …*. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-35758-9_6

Nonaka, I., & Takeuchi, H. (1995). *The Knowledge-creating Company: How Japanese Companies Create the Dynamics of Innovation* (p. 284). Oxford University Press. Retrieved from http://books.google.com/books?hl=en&lr=&id=B-qxrPaU1-MC&pgis=1

Padhy, N., Mishra, D. P., & Panigrahi, R. (2012). The Survey of Data Mining Applications And Feature Scope. Retrieved from http://arxiv.org/abs/1211.5723

Patel-Schneider, P. (1989). Undecidability of subsumption in NIKL. *Artificial Intelligence*. Retrieved from http://www.sciencedirect.com/science/article/pii/0004370289900301

Patton, J. (2008). User Story Mapping. Retrieved from http://www.cs.northwestern.edu/academics/courses/394/slides/spr12/Patton Building Better Products Using.pdf

Pearson, K. (1895). Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of …*. Retrieved from http://rspl.royalsocietypublishing.org/content/58/347-352/240.full.pdf

Plackett, R. (1983). Karl Pearson and the chi-squared test. *… Statistical Review/Revue Internationale de Statistique*. Retrieved from http://www.jstor.org/stable/1402731

Power, D. J. (2007). A Brief History of Decision Support Systems. *DSSResources.COM*. Retrieved from http://www.mendeley.com/research/brief-history-decision-support-systems-47/

Quinlan, J. (1986). Induction of decision trees. *Machine Learning*. Retrieved from http://link.springer.com/article/10.1023/A:1022643204877

Quinlan, J. (1993). *C4. 5: programs for machine learning*. Retrieved from http://books.google.ch/books?hl=en&lr=&id=HExncpjbYroC&oi=fnd&pg=PR7&dq=Quinlan,+J.+R.+(1993).+C4.5:+Programs+for+Machine+Learning&ots=nLl9iXq0Xr&sig=58TLGyNrVwJh251XcdcJDhAO7nI

Rokach, L. (2008). *Data mining with decision trees: theory and applications*. Retrieved from http://books.google.ch/books?hl=en&lr=&id=GlKIIR78OxkC&oi=fnd&pg=PR7&dq=Rokach,+Lior.+2008.+Data+mining+with+decision+trees:+theory+and+applications&ots=0-ZIjR_c_R&sig=HO0Y4T0P-fmW_OxS97hKJ7LW_1U

Schölkopf, B., & Smola, A. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Retrieved from http://books.google.ch/books?hl=en&lr=&id=y8ORL3DWt4sC&oi=fnd&pg=PR13&dq=support+vector+machine+rbf+kernel&ots=bKuRbyWbIH&sig=djiz3YT1rtrw3wDMC6BM4nGXAM8

Skarka, W. (2007). Application of MOKA methodology in generative model creation using CATIA. *Engineering Applications of Artificial Intelligence*, *20*(5), 677–690. doi:10.1016/j.engappai.2006.11.019

Sliger, M. (2006). Agile Projects in the Waterfall Enterprise. *Better Software*. Retrieved from http://programmepmo.programmedevelopment.com/public/uploads/files/agile_-_bridging_the_gap.pdf

Solomatine, D., & Shrestha, D. (2004). AdaBoost. RT: a boosting algorithm for regression problems. *Neural Networks, 2004. …*. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1380102

Staab, S., & Studer, R. (Eds.). (2004). *Handbook on Ontologies*. Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-540-24750-0

Stark, J. (2005a). Maturity model for PDM. *Product Lifecycle Management: 21st Century Paradigm …*. Retrieved from http://link.springer.com/content/pdf/10.1007/1-84628-067-2_29.pdf

Stark, J. (2005b). PDM, an essential enabler for PLM. *Product Lifecycle Management: 21st Century Paradigm …*. Retrieved from http://link.springer.com/content/pdf/10.1007/1-84628-067-2_22.pdf

Stewart, T., & Ruckdeschel, C. (1998). Intellectual capital: The new wealth of organizations. *Performance Improvement*, *37*(7), 56–59. doi:10.1002/pfi.4140370713

Suárez-Figueroa, M. C. (2010, June 25). NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse. Facultad de Informática (UPM). Retrieved from http://oa.upm.es/3879/2/MARIA_DEL-_CARMEN_SUAREZ_DE_FIGUEROA_BAONZA.pdf

Sun, T. (2003). Decomposition methods for linear support vector machines. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).* (Vol. 4, pp. IV–868–71). IEEE. doi:10.1109/ICASSP.2003.1202781

Sveiby, K. E. (1997). *The New Organizational Wealth: Managing & Measuring Knowledge-based Assets* (p. 220). Berrett-Koehler Publishers. Retrieved from http://books.google.com/books?hl=en&lr=&id=xKNXlgaeCjAC&pgis=1

Tan, P., & Dowe, D. (2005). MML inference of oblique decision trees. *AI 2004: Advances in Artificial Intelligence*. Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-30549-1_105

Terzi, S., Bouras, A., Dutta, D., Garetti, M., & Kiritsis, D. (n.d.). Product lifecycle management - from its history to its new role. *International Journal of Product Lifecycle Management*, *4*(4), 360–389. Retrieved from http://cat.inist.fr/?aModele=afficheN&cpsidt=23501572

Thomas Eiter, G. I. A. P. R. S. H. T. (n.d.). Reasoning with rules and ontologies. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.103.2811

Vapnik, V. (2000). *The nature of statistical learning theory*. Retrieved from http://books.google.ch/books?hl=en&lr=&id=sna9BaxVbj8C&oi=fnd&pg=PR7&dq=Vapnik,+V.+N.+(1995).+The+nature+of+statistical+learning+theory.+Springer&ots=ooI8NYosdd&sig=tfpk_RqnWj81p-joWNZjUyuQJ84

Wiig, K. M. (1997). Knowledge management: Where did it come from and where will it go? *Expert Systems with Applications*, *13*(1), 1–14. doi:10.1016/S0957-4174(97)00018-3

Wu, L., Barash, G., & Bartolini, C. (2007). A Service-oriented Architecture for Business Intelligence. In *IEEE International Conference on Service-Oriented Computing and Applications (SOCA '07)* (pp. 279–285). IEEE. doi:10.1109/SOCA.2007.6

Yuan, S.-T., & Sun, J. (2004). Ontology-Based Structured Cosine Similarity in Speech Document Summarization, 508–513. doi:10.1109/WI.2004.110

Zhong, P., & Fukushima, M. (2007). Regularized nonsmooth Newton method for multi-class support vector machines. *Optimisation Methods and Software*. Retrieved from http://www.tandfonline.com/doi/abs/10.1080/10556780600834745

Appendix A SWRL rules for industrial standards ontology

1. Alarm(?l), AsDesignedMainWalk(?a), StandardMainWalk(?b), HasHorizontalLength(?a, ?c), HasHorizontalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check horizontal length of Main Walkways”)

2. Alarm(?l), AsDesignedMainWalk(?a), StandardMainWalk(?b), HasVerticalLength(?a, ?c), HasVerticalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check vertical length of Main Walkways”)

3. Alarm(?l), AsDesignedAccess(?a), StandardAccess(?b), HasHorizontalLength(?a, ?c), HasHorizontalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check horizontal length of Access Ways”)

4. Alarm(?l), AsDesignedAccess(?a), StandardAccess(?b), HasVerticalLength(?a, ?c), HasVerticalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check vertical length of Access Ways”)

5. Alarm(?l), AsDesignedHatch(?a), StandardHatch(?b), HasHorizontalLength(?a, ?c), HasHorizontalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check horizontal length of Hatch openings”)

6. Alarm(?l), AsDesignedHatch(?a), StandardHatch(?b), HasVerticalLength(?a, ?c), HasVerticalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check vertical length of Hatch openings”)

7. Alarm(?l), AsDesignedTransportation(?a), StandardTransportation(?b), HasVerticalLength(?a, ?c), HasVerticalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check vertical length of Transportation Ways”)

8. Alarm(?l), AsDesignedWorkAreas(?a), StandardWorkAreas(?b), HasVerticalLength(?a, ?c), HasVerticalLength(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check vertical length of Work Areas”)

9. Alarm(?l), AsDesignedFlangeOnPipe(?a), StandardFlangeOnPipe(?b), HasDiameter(?c, ?a), HasDiameter(?d, ?b), greaterThanOrEqual(?c, ?d), DistanceToFixedObstruction(?e, ?a), DistanceToFixedObstruction(?f, ?b), lessThanOrEqual(?e, ?f) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Check distance between Flange and Fixed Obstruction”)

10. Alarm(?l), AsDesignedCabinet(?a), StandardCabinet(?b), Floor(?r), HasHeight(?a, ?c), HasHeight(?b, ?d), HasHeight(?r, ?s), subtract(?e, ?d, ?s), subtract(?f, ?c, ?s), greaterThan(?f, ?e) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check height of Cabinet”)

11. Alarm(?l), AsDesignedSeated(?a), StandardSeated(?b), HasHeight(?a, ?c), HasMinHeight(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Table Top Height for seated work is too low”)

12. Alarm(?l), AsDesignedSeated(?a), StandardSeated(?b), HasHeight(?a, ?c), HasMaxHeight(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Table Top Height for seated work is too high”)

13. Alarm(?l), AsDesignedSeated(?a), StandardSeated(?b), HasThickness(?a, ?c), HasThickness(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check thickness of Table Top for Seated work”)

14. Alarm(?l), AsDesignedSeated(?a), StandardSeated(?b), HasWidthForLegs(?a, ?c), HasWidthForLegs(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Check Width for legs below work surface for Seated work”)

15. Alarm(?l), AsDesignedSeated(?a), StandardSeated(?b), HasDepthForLegs(?a, ?c), HasDepthForLegs(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Check Depth for legs below work surface for Seated work”)

16. Alarm(?l), AsDesignedNormal(?a), StandardNormal(?b), HasHeight(?a, ?c), HasMinHeight(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Table Top Height for Normal standing work is too low”)

17. Alarm(?l), AsDesignedNormal(?a), StandardNormal(?b), HasHeight(?a, ?c), HasMaxHeight(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Table Top Height for Normal standing work is too high”)

18. Alarm(?l), AsDesignedClassic(?a), StandardClassic(?b), HasHeight(?a, ?c), HasMinHeight(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Classic Workbench Height is too low”)

19. Alarm(?l), AsDesignedClassic(?a), StandardClassic(?b), HasHeight(?a, ?c), HasMaxHeight(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Classic Workbench Height is too high”)

20. Alarm(?l), AsDesignedGalley(?a), StandardGalley(?b), HasHeight(?a, ?c), HasMinHeight(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Galley Workbench Height is too low”)

21. Alarm(?l), AsDesignedGalley(?a), StandardGalley(?b), HasHeight(?a, ?c), HasMaxHeight(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Galley Workbench Height is too high”)

22. Alarm(?l), AsDesignedWbVDU(?a), StandardWbVDU(?b), HasHeight(?a, ?c), HasMinHeight(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “VDU Workbench Height is too low”)

23. Alarm(?l), AsDesignedWbVDU(?a), StandardWbVDU(?b), HasHeight(?a, ?c), HasMaxHeight(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “VDU Workbench Height is too high”)

24. Alarm(?l), AsDesignedVDU(?a), StandardVDU(?b), HasHeight(?a, ?c), HasMinHeight(?b, ?d), lessThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Table Top Height for VDU standing work is too low”)

25. Alarm(?l), AsDesignedVDU(?a), StandardVDU(?b), HasHeight(?a, ?c), HasMaxHeight(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “Table Top Height for VDU standing work is too high”)

26. Alarm(?l), AsDesignedCD(?a), StandardCD(?b), HasHeight(?a, ?c), HasHeight(?b, ?d), greaterThan(?c, ?d) -> isActive(?l, true), Address(?l, “designer”), Text(?l, “check Control Device height”)

Appendix B SWRL rules for design recommendation ontology

1. Customer(?c), AsDesignedProduct(?a), Alarm(?l), hasAvailability(?c, ?x), hasAvailability(?a, ?y), greaterThan(?x, ?y) -> isActive(?l, true), Text(?l, “Check Customer Required Availability”), Address(?l, “Designer”)
2. Customer(?c), AsDesignedProduct(?a), Alarm(?l), hasQualityIndex(?c, ?x), hasQualityIndex(?a, ?y), greaterThan(?x, ?y) -> isActive(?l, true), Text(?l, “Check Customer Required Quality Index”), Address(?l, “Designer”)
3. Customer(?c), AsDesignedProduct(?a), Alarm(?l), hasDownTime(?c, ?x), hasDownTime(?a, ?y), lessThan(?x, ?y) -> isActive(?l, true), Text(?l, “Check Customer Requested DownTime”), Address(?l, “Designer”)
4. Customer(?c), AsDesignedProduct(?a), Alarm(?l), hasSurface(?c, ?x), hasSurface(?a, ?y), lessThan(?x, ?y) -> isActive(?l, true), Text(?l, “Check Customer Requested Surface”), Address(?l, “Designer”)
5. Customer(?c), AsDesignedProduct(?a), Alarm(?l), hasCycleT(?c, ?x), hasCycleT(?a, ?y), lessThan(?x, ?y) -> isActive(?l, true), Text(?l, “Check Customer Requested Cycle Time”), Address(?l, “Designer”)
6. Customer(?c), Alarm(?l), hasWCM(?c, ?x), equal(?x, 1) -> isActive(?l, true), Text(?l, “Check Customer Requested WCM”), Address(?l, “Designer”)
7. AsDesignedProduct(?a), hasAvailability(?a, ?u), MTBF(?a, ?v), MTTR(?a, ?w), divide(?x, ?v, ?y), add(?y, ?v, ?w) -> equal(?u, ?x)
8. AsDesignedProduct(?a), AsManufacturatedProduct(?m), hasProductivity(?a, ?s), has Availability(?a, ?t), hasCycleT(?a, ?u), hasWorkTime(?m, ?v), divide(?w, ?x, ?u), multiply(?x, ?y, 3600), multiply(?y, ?v, ?t) -> equal(?s, ?w)
9. AsManufacturatedProduct(?m), Alarm(?l), MTBF(?m, ?x), hasWorkTime(?m, ?y), equal(?x, ?z), divide(?z, ?y, 10) -> isActive(?l, true), Text(?l, “The line is not operating properly”), Address(?l, “Servicing”)
10. AsManufacturatedProduct(?m), Alarm(?l), MTTR(?m, ?x), greaterThan(?x, 7,5) -> isActive(?l, true), Text(?l, “The line is not operating properly”), Address(?l, “Servicing”)
11. Customer(?c), Alarm(?l), hasRetoolBOL(?c, ?x), equal(?x, 1) -> isActive(?l, true), Text(?l, “Customer Requires retool able machine”), Address(?l, “Designer”)
12. AsDesignedProduct(?a), AsManufacturatedProduct(?m), Alarm(?l), hasAvailability(?a, ?x), hasAvailability(?m, ?y), notEqual(?x, ?y) -> isActive(?l, true), Text(?l, “Update availability database”), Address(?l, “expert”)

13.) AsDesignedProduct(?a), AsManufacturatedProduct(?m), Alarm(?l), hasProductivity(?a, ?x), hasProductivity(?m, ?y), notEqual(?x, ?y) -> isActive(?l, true), Text(?l, “Update productivity database”), Address(?l, “expert”)
14. Customer(?c), AsDesignedProduct(?a), AsDesignedPart(?p), Alarm(?l), hasCycleT(?c, ?x), hasCycleT(?a, ?y), lessThan(?x, ?y), hasSurface(?c, ?i), hasSurface(?a, ?j), hasSurface(?p, ?k), greaterThan(?i, ?z), add(?z, ?j, ?k) -> isActive(?l, true), Text(?l, “Consider adding a part to shorten cycle time”), Address(?l, “Designer”)
15. Customer(?c), AsDesignedProduct(?a), AsDesignedPart(?p), Alarm(?l), hasAvailability(?c, ?x), hasAvailability(?a, ?y), greaterThan(?x, ?y), hasSurface(?c, ?i), hasSurface(?a, ?j), hasSurface(?p, ?k), greaterThan(?i, ?z), add(?z, ?j, ?k) -> isActive(?l, true), Text(?l, “Consider adding a part to shorten cycle availability”), Address(?l, “Designer”)

Appendix C SWRL rules for knowledge management ontology

1. Employee(?x), Employee(?y), K-Asset(?z), isOwnedBy(?z, ?x), isReviewedBy(?z, ?y), EmployeeName(?y, ?t), Superior(?x, ?t) -> KnowledgeMaturity(?z, 3)
2. K-Asset(?z), Reviewed(?z, true) -> KnowledgeMaturity(?z, 1)
3. Employee(?x), Employee(?y), K-Asset(?z), isOwnedBy(?z, ?x), isReviewedBy(?z, ?y), EmployeeName(?y, ?t), Superior(?x, ?t) -> KnowledgeMaturity(?z, 2)
4. Alert(?d), Discipline(?b), Product(?s), Project(?p), DisciplineName(?b, "Structure"), Location(?p, "NCS"), ProductName(?s, "Topside") -> Active(?d, true), Recipient(?d, "Designer"), Text(?d, "Recomended standard EUROCODE3 ")
5. Alert(?d), Discipline(?b), Employee(?x), Product(?a), Project(?c), DisciplineName(?b, "Structure"), EmployeeDiscipline(?x, "Designer"), Location(?c, "NCS"), ProductName(?a, "Stairtower") -> Active(?d, true), Recipient(?d, ?x), Text(?d, "NORSOK standard S-002 Working Environment (provided)")
6. Alert(?d), Discipline(?b), Employee(?x), Product(?a), Project(?c), DisciplineName(?b, "Structure"), EmployeeDiscipline(?x, "Designer"), Location(?c, "Asia"), ProductName(?a, "Topside") -> Active(?d, true), Recipient(?d, ?x), Text(?d, " Recommended standard is API/AISC")
7. Alert(?d), Discipline(?b), Employee(?x), Product(?a), Project(?c), DisciplineName(?b, "Structure"), EmployeeDiscipline(?x, "Designer"), Location(?c, "Asia"), ProductName(?a, "Stairtower") -> Active(?d, true), Recipient(?d, ?x), Text(?d, " Recommended standard is AS 1657 - 2013")

Appendix D SWRL rules for quality control ontology

1. As_Measured(?m), Heating(?h), Hot_Stamping(?s), Optimal(?o), hasOvenTemperature(?h, ?y), hasOvenTemperature(?o, ?x), greaterThan(?z, 20), subtract(?z, ?x, ?y) -> ConsideredRisky(?m, true)
2. Deviation(?d), Mechanical(?m), Quality_aspect(?q), isPresent(?m, false) -> CAD_as_produced(?c), Defect(?c, false)
3. As_Measured(?m), Hot_Stamping(?h), Optimal(?o), Pressing(?p), hasInsertionTemperature(?m, ?x), hasInsertionTemperature(?o, ?y), lessThan(?y, ?x) -> Deviation(?d), Mechanical(?k), Quality_aspect(?q), isPresent(?k, false)
4. As_Measured(?m), Heating(?h), Optimal(?o), hasDwellTime(?h, ?x), hasDwellTime(?o, ?y), greaterThan(?x, ?y) -> ConsideredRisky(?m, true)
5. CAD_as_produced(?c), Quality_aspect(?q), Structural(?s), ROIFound(?s, ?r), stringEqualIgnoreCase(?r, "0") -> Defect(?c, true)
6. CAD_as_produced(?c), Scanning(?s), Performed(?s, false) -> isPresent(?c, false)
7. As_Measured(?m), Hot_Stamping(?h), Optimal(?o), Pressing(?p), hasInsertionTemperature(?m, ?x), hasInsertionTemperature(?o, ?y), lessThan(?x, ?y) -> Deviation(?d), Mechanical(?k), Quality_aspect(?q), isPresent(?k, true)
8. As_Measured(?m), Heating(?h), hasCycleNumber(?m, ?c), lessThan(?c, 20) -> isInTransientState(?h, false)
9. As_Measured(?m), Optimal(?o), Pressing(?p), hasInsertionTemperature(?o, ?x), hasInsertionTemperature(?p, ?y), lessThan(?z, -20), subtract(?z, ?x, ?y) -> ConsideredRisky(?m, true)
10. CAD_as_produced(?c), Microstructural(?m), Quality_aspect(?q), hasMeasuredValue(?m, ?x), hasTolerance(?m, ?y), greaterThan(?x, ?y) -> Defect(?c, true)
11. As_Measured(?m), Optimal(?o), Pressing(?p), hasPressingForce(?o, ?x), hasPressingForce(?p, ?y), greaterThan(?z, 50000), subtract(?z, ?x, ?y) -> ConsideredRisky(?m, true)
12. CAD_as_produced(?c), Defect(?c, false) -> Condition_Error(?e), hasConditionError(?e, false)
13. As_Measured(?m), Optimal(?o), Pressing(?p), hasCycleTime(?o, ?x), hasCycleTime(?p, ?y), greaterThan(?z, 2), subtract(?z, ?x, ?y) -> ConsideredRisky(?m, true)

14. As_Measured(?m), Scanning(?s), ConsideredRisky(?m, true) -> Performed(?s, true)
15. As_Measured(?m), Optimal(?o), Pressing(?p), hasBlankHoldForce(?o, ?x), hasBlankHoldForce(?p, ?y), greaterThan(?z, 20), subtract(?z, ?y, ?x) -> ConsideredRisky(?m, true)
16. Deviation(?d), Mechanical(?m), Quality_aspect(?q), isPresent(?m, true) -> CAD_as_produced(?c), Defect(?c, true)
17. CAD_as_produced(?c), Defect(?c, true) -> Condition_Error(?e), hasConditionError(?e, true)
18. CAD_as_produced(?c), Scanning(?s), Performed(?s, true) -> isPresent(?c, true)
19. As_Measured(?m), Pressing(?p), hasPositioning(?p, true) -> Setup_Error(?e), hasSetupError(?e, true)
20. As_Measured(?m), Optimal(?o), Pressing(?p), hasExtractionTemperature(?o, ?x), hasExtractionTemperature(?p, ?y), greaterThan(?z, 20), subtract(?z, ?x, ?y) -> ConsideredRisky(?m, true)
21. As_Measured(?m), Heating(?h), hasCycleNumber(?m, ?c), lessThan(20, ?c) -> isInTransientState(?h, true)
22. As_Measured(?m), CAD_as_produced(?c), Deviation(?d), Geometric(?g), Hot_Stamping(?h), Optimal(?o), Pressing(?p), hasExtractionTemperature(?o, ?t), hasExtractionTemperature(?p, ?f), hasMeasuredValue(?g, ?x), hasTolerance(?g, ?y), greaterThan(?f, ?t), greaterThan(?x, ?y) -> Tool(?l), hasCorrectGeometry(?l, false), hasPositioning(?p, true)

Appendix E SWRL rules for predictive maintenance ontology

1. Filers(?f), Filer_Change(?fc), expiration(?f, ?ff), frequency(?fc, ?fcf), lessThen(?ff, ?fcf), Alert(?a) -> message(?a, "Change the filters")
2. Oil(?o), Oil_Change(?oc), expiration(?o, ?of), frequency(?oc, ?ocf), lessThen(?of, ?ocf), Alert(?a) -> message(?a, "Change the oil")
3. Conditions(?c), Avtemp(?c, ?x), Oil(?o), expiration(?o, ?e), greaterThan(?x, 30), multiply(?z, ?e, 0.8) -> expiration(?o, ?z)
4. Conditions(?c), Avpollution(?c, ?x), Oil(?o), expiration(?o, ?e), greaterThan(?x, 200), multiply(?z, ?e, 0.8) -> expiration(?o, ?z)
5. Conditions(?c), AvRpM(?c, ?x), Oil(?o), expiration(?o, ?e), greaterThan(?x, 4000), multiply(?z, ?e, 0.8) -> expiration(?o, ?z)
6. Oil_change(?oc), Engine(?e), Ba(?oc, ?x), greaterThan(?x,50) ->running(?e, false)
7. Oil_change(?oc), Engine(?e), Ca(?oc, ?x), greaterThan(?x, 50) ->running(?e, false)
8. Oil_change(?oc), Engine(?e), Mg(?oc, ?x), greaterThan(?x, 20) ->running(?e, false)
9. Oil_change(?oc), Engine(?e), P(?oc, ?x), greaterThan(?x, 5) ->running(?e, false)
10. Oil_change(?oc), Engine(?e), Zn(?oc, ?x), greaterThan(?x, 5) ->running(?e, false)
11. Oil_change(?oc), Engine(?e), K(?oc, ?x), greaterThan(?x, 20) ->running(?e, false)
12. Oil_change(?oc), Engine(?e), Si(?oc, ?x), greaterThan(?x, 45) ->running(?e, false)
13. Oil_change(?oc), Engine(?e), Na(?oc, ?x), greaterThan(?x, 40) ->running(?e, false)
14. Oil_change(?oc), Engine(?e), Al(?oc, ?x), greaterThan(?x, 15) ->running(?e, false)
15. Oil_change(?oc), Engine(?e), Cr(?oc, ?x), greaterThan(?x, 8) ->running(?e, false)
16. Oil_change(?oc), Engine(?e), Cu(?oc, ?x), greaterThan(?x, 120) ->running(?e, false)
17. Oil_change(?oc), Engine(?e), Fe(?oc, ?x), greaterThan(?x, 120) ->running(?e, false)
18. Oil_change(?oc), Engine(?e), Pb(?oc, ?x), greaterThan(?x, 25) ->running(?e, false)
19. Oil_change(?oc), Engine(?e), Ni(?oc, ?x), greaterThan(?x, 5) ->running(?e, false)
20. Oil_change(?oc), Engine(?e), Ag(?oc, ?x), greaterThan(?x, 3) ->running(?e, false)
21. Oil_change(?oc), Engine(?e), Sn(?oc, ?x), greaterThan(?x, 8) ->running(?e, false)
22. Oil_change(?oc), Engine(?e), Ti(?oc, ?x), greaterThan(?x, 3) ->running(?e, false)

Curriculum Vitae

Ana Milicic

Date of birth: 05.10.1984

Place of birth: Novi Sad, Serbia

Address: Rue Marterey 26

1005 Lausanne, Switzerland

Phone num: +41 76 642 67 14

E-mail: ana.milicic.ns@gmail.com

Education

- **2011. – present**
  Enrolled in PhD program at EPFL, Switzerland. Program involves knowledge management approaches for product lifecycle and technologies to support it.

- **2009. – 2011.**
  Master track program at Temple University, USA. Program includes advanced courses in artificial intelligence, data mining, multimedia data processing, analysis of algorithms and related topics.

- **2003. – 2009.**
  M.Sc. Microcomputer Electronics from University of Novi Sad, Serbia. Program involves design of electronics systems, algorithms for system optimization and signal processing.

Work experience

2011-present : Doctoral assistant at EPFL :

- **Team member** in European project LinkeDesign, involving several industrial partners, such as Volkswagen and Comau. Outcome is software platform Leap for gathering, structuring and exploitation of various data sources. Leap includes tools for decision support and context dependent alert system for all involved actors. . (2011-2015)

- **Project leader** of EPFL and Liebherr company joint project. Algorithm for predictive maintenance of diesel engines was developed. Domain analysis resulted in collection of sensor data, engine profiling and engine oil analysis data. (2014.)
- **Project leader** of EPFL and Aker company joint project. Delivered was a solution for experts knowledge structuring and knowledge base maintenance mechanisms. Concept of K-brief was exploited to create interactive user interface with easy retrieval of context relevant information. (2014.)

2009-2011 : Research assistant at Temple University:

- **Team member** of Temple University, research project involving social network user profiling and user content popularity prediction. (2012.)

Languages

- English: Proficient
- French: Basic understanding
- Serbian: Native

Publications

International journals papers :

Milicic, A., El Kadiri, S., Perdikakis, A., Ivanov, P., & Kiritsis, D. (2014). Toward the definition of domain concepts and knowledge through the application of the user story mapping method. *International Journal of Product Lifecycle Management*, *7*(1), 3-16.

Milicic, A., El Kadiri, S., & Kiritsis, D. (2015). (FoF) Autonomous system for PLM domain data exploitation" .*International Journal of Computer Integrated Manufacturing (accepted)*

Conference papers :

Cerri D., Taisch M., Terzi S., Buda A., Framling K.,El Kadiri S., Milicic A., Kiritsis D, Parrotta S., Peukert E. (2015). Proposal of a Closed Loop Framework for the Improvement of Industrial Systems' Life Cycle Performance: Experiences from the LinkedDesign Project. In *The 22nd CIRP conference on Life Cycle Engineering, At Sydney, Australia*

Milicic A. (2014). Automated analysis of product related data generated from PLM ontology. In *Doctoral Workshop on "Product and Asset Lifecycle Management", Les Diablerets, Switzerland*

El Kadiri, S., Milicic, A., & Kiritsis, D. (2013). Linked Data Exploration in Product Life-Cycle Management. In *Advances in Production Management Systems. Sustainable Production and Service Supply Chains* (pp. 460-467). Springer Berlin Heidelberg.

Kiritsis, D., El Kadiri, S., Perdikakis, A., Milicic, A., Alexandrou, D., & Pardalis, K. (2013). Design of fundamental ontology for manufacturing product lifecycle applications. In *Advances in Production Management Systems. Competitive Manufacturing for Innovative Products and Services* (pp. 376-382). Springer Berlin Heidelberg.

Milicic, A., Perdikakis, A., El Kadiri, S., & Kiritsis, D. (2013, June). PLM Ontology Exploitation through Inference and Statistical Analysis: Case Study for LCC. In *Manufacturing Modelling, Management, and Control* (Vol. 7, No. 1, pp. 1004-1008).

Milicic, A., Perdikakis, A., El Kadiri, S., Kiritsis, D., Terzi, S., Fiordi, P., & Sadocco, S. (2012, January). Specialization of a Fundamental Ontology for Manufacturing Product Lifecycle Applications: A Case Study for Lifecycle Cost Assessment. In *On the Move to Meaningful Internet Systems: OTM 2012 Workshops* (pp. 69-72). Springer Berlin Heidelberg.

Milicic, A., Perdikakis, A., El Kadiri, S., & Kiritsis, D.. "Towards the Definition of Domain Concepts and Knowledge through the Application of the User Story Mapping Method." *Product Lifecycle Management. Towards Knowledge-Rich Enterprises*. Springer Berlin Heidelberg, 2012. 58-69.