INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Ontologias para Manutenção Preditiva com Dados sensíveis ao tempo

ARMANDO JORGE VENTURA NOBRE
Outubro de 2022

POLITÉCNICO
DO PORTO

Ontologies for Predictive Maintenance with Time-Sensitive Data

Armando Jorge Ventura Nobre

Master's dissertation in
Informatics Engineering, Area of Specialization in
Graphic Systems and Multimedia

Advisor: Goreti Marreiros
Co-advisor: Alda Canito

Porto, October 2022

Dedication

This thesis is dedicated to my wife Laura and daughters, Jessica and Deborah, who always supported me in the pursuit of knowledge. They provided me with the necessary stamina to demonstrate that no one is too old to learn, and once a goal is achieved, the achievement itself is greater than any prize. And nothing, but nothing equals the reward of enlightenment.

Resumo

As empresas de fabrico industrial devem assegurar um processo produtivo contínuo para serem competitivas e fornecer os produtos fabricados no prazo e com a qualidade exigida pelos clientes. A quebra da cadeia de fabrico pode ter desfechos graves, resultando numa redução da produção e na interrupção da cadeia de abastecimento. Estes processos são compostos por cadeias de máquinas que executam tarefas em etapas. Cada máquina tem uma tarefa específica a executar, e o resultado de cada etapa é fornecido à próxima etapa. Uma falha imprevista numa das máquinas tende a interromper toda a cadeia produtiva.

A manutenção preventiva agendada tem como objetivo evitar a ocorrência de falhas, tendo como base o tempo médio antes da falha (MTBF), que representa a expectativa média de vida de componentes individuais com base em dados históricos. As tarefas de manutenção podem implicar um período de paralisação e a interrupção da produção. Esta manutenção é executada rotineiramente e a substituição de componentes não considera a necessidade premente da sua substituição, sendo os mesmos substituídos com base no ciclo do agendamento.

É aqui que a manutenção preditiva é aplicável. Efetuando a recolha de dados de sensores dos equipamentos, é possível detetar irregularidades nos dados recolhidos, através da aplicação de processos de raciocínio e inferência, conduzindo à atempada previsão e deteção de falhas. Levando este cenário à otimização do tempo de manutenção, evitando falhas inesperadas, à redução de custos e ao aumento da produtividade em comparação com a manutenção preventiva. Os dados fornecidos pelos sensores são sensíveis ao tempo, variações e flutuações ocorrem ao longo do tempo e devem ser analisados em relação ao período em que ocorrem.

Esta dissertação tem como objetivo o desenvolvimento de uma ontologia para a manutenção preditiva que descreva a sua abrangência e o campo da sua aplicação. A aplicabilidade da ontologia será demonstrada com uma ferramenta, igualmente desenvolvida, que transforma dados sensíveis ao tempo recolhidos em tempo real a partir de sensores de máquinas industriais, fornecidos por WebServices, em indivíduos dessa mesma ontologia, considerando a representação do fator temporal dos dados.

Palavras-chave: Ontologias, Manutenção preditiva, Dados sensíveis ao tempo, Raciocínio temporal, Dados de sensores

Abstract

Manufacturing companies must ensure a continuous production process to be competitive and supply the manufactured goods in time and with the desired quality the customers expect. Any disruption in the manufacturing chain may have disastrous consequences, representing a shortage of production and the interruption of the supply chain. The manufacturing processes are composed of a chain of industrial machines operating in stages. Each machine has a specific task to complete, and the result of each stage is forwarded to the next stage. An unpredicted malfunction of one of the machines tends to interrupt the whole production chain.

Scheduled Preventive maintenance intends to avoid causes leading to faults, but relies on parameters such as Mean Time Before Failure (MTBF), which represents the average expected life span of individual components based on statistical data. A maintenance task may lead to a period of downtime and consequently to a production halt. Being the maintenance scheduled and executed routinely, the replacement of components, does not consider the effective need of its replacement, they are replaced based on the scheduling cycle.

This is where predictive maintenance is applicable. By collecting sensor data of industrial equipment, anomalies can be determined through reasoning and inference processes applied to the data, leading to an early fault and time to failure prediction. This scenario leads to maintenance timing optimization, avoidance of unexpected failures, cost savings and improved productivity when compared to preventive maintenance. Data supplied by sensors is time-sensitive, as variations and fluctuations occur over periods of time and must be analysed concerning the period they occur.

This dissertation aims to develop an ontology for predictive maintenance that describes the scope and field of application. The applicability of the ontology will be demonstrated with a tool, also to be developed, that transforms time-sensitive data collected in real time from sensors of industrial machines, provided by a WebServices, into individuals of the same ontology, considering the representation of the temporal factor of the data.

Keywords: Ontologies, Predictive Maintenance, Time-Sensitive Data, Temporal Reasoning, Sensor Data

Acknowledgements

I wish to express my gratefulness towards my advisor and professor, Goreti Marreiros, who provided me some years ago a challenging and passionate subject for my Bachelor's Thesis, and now, at my renewed request, for my Master's Dissertation, once again, an even more challenging and enthusiastic subject for my academic work.

I also would like to express my utmost gratitude to my co-advisor, Alda Canito, who for the second time (firstly my Bachelor's and secondly my Master's Dissertation) coached and guided me through this endeavouring world of ontologies. I'm profoundly convinced that her coaching, inspiration, and guidance was a fundamental driver during all those months of investigation and academic work, allowing for the conclusion of this dissertation. As my co-advisor, she always encouraged me to address the most complex problems, as if they were the simplest questions in the world to be answered. This encouragement fuelled my motivation for this present work.

I must address a very special acknowledgement and gratitude, not to a single person, but the whole faculty of ISEP. No words can express enough gratefulness to those who have chosen to dedicate their lives conveying knowledge. It was through this faculty, that ISEP as an academic institution changed my life, from the moment I entered the main entrance as a freshman student.

ISEP never ceased to be passionate in my endless craving for new knowledge, since the first second I sat down in the front row of classroom B301, at 18:10, the 14th of September 2015.

Table of Contents

List of Figures

List of Tables

Acronyms and Symbols

List of Acronyms

CMMS Computerized maintenance management system

CSV Comma Separates Value

DM Data Mining

e.g. *exempli gratia*

ERP Enterprise Resource Planning

FEI Front End of Innovation

GECAD Grupo de Investigação em Engenharia e Computação Inteligente para a Inovação e o Desenvolvimento

GFO General Formal Ontology

GRASP General Responsibility Assignment Software Pattern

HTTP Hypertext Transfer Protocol

Hz Hertz

IoT Internet of Things

JIT Just-in-Time

JSON JavaScript Object Notation

kg/cm^2 Kilogram per square centimetre

lb/ft^2 Pound per square foot

ML Machine Learning

MTBF Mean Time Before Failure

MRP Manufacturing Resource Planning

NCD New Concept Development

NPPD New Product Process Development

OD101	Ontology Development 101
OOPS!	OntOlogy Pitfall Scanner!
OWL	Web Ontology Language
QFD	Quality Function Deployment
SSD	System Sequence Diagrams
SSN	Semantic Sensor Network
PIANiSM	Predictive and Prescriptive Automation Smart Manufacturing
PdM	Predictive Maintenance
POM	Project Object Model
PvM	Preventive Maintenance
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SPARQL	Simple Protocol and Rdf Query Language
T-Index	Time-Indexed
TOPSIS	Technique for Order Preference by Similarity to Ideal Solution
UUID	Universally Unique Identifier (UUID)
VPC	Value Proposition Canvas
W3C	World Wide Web Consortium
WWW	World Wide Web
XML	eXtensible Markup Language

1 Introduction

This document systematizes the work developed during this Master's Dissertation in Informatics Engineering at *Instituto Superior de Engenharia do Porto* (ISEP).

This work was developed in the context of a collaborative international project: Predictive and Prescriptive Automation in Smart Manufacturing (PIANiSM, 2020), whose Portuguese partners include the *Instituto Superior de Engenharia do Porto (ISEP), Sistrade Software Consulting* and *Vizelpas – Comércio de Artigos Plásticos LDA*. The purposes of this work were two-fold: the development of an ontology for the Predictive Maintenance (PdM) in the plastic extrusion domain, and a tool that facilitates the transformation of data acquired by sensors and other contextual information into individuals of said ontology, such that it conveniently represents its time-sensitive nature in a way that can be exploited by predictive algorithms.

In this chapter, a proper context for the dissertation is presented and a description of the problem to be solved is made. Also, the objectives related to this work are hereby related, as well as the approaches and development process followed. This chapter is concluded with an overview of the structure of the present document, summarizing the subjects covered in each chapter.

1.1 Context

Maintenance is a fundamental activity in industrial manufacturing processes, preventing equipment failures and avoiding recurrent downtime periods where worn-out components need replacement. This need emerges from the high demands of today's industry to deliver manufactured goods at a continuous pace. Just-in-Time (JIT) production is globally adopted for reducing times within production systems and leads to the best response times in terms of delivery from suppliers to customers (Kootanaee, Babu, & Talari, 2013). To keep up with effective JIT production, unexpected equipment maintenance must be reduced, and failures should be avoided. Particularly those failures resulting in unpredicted downtime, which may

lead to interruptions in production and potential catastrophic scenarios that no manufacturing company wishes to endure.

While many manufacturing companies are still adopting traditional reactive or preventive maintenance, the opportunity to follow the modernization path with the introduction of Internet of Things (IoT) technologies reached a stage of maturity, leading to a widened acceptance. The industrial transformation permitting the communication from machine to machine enabled by the IoT represents Industry 4.0, the Fourth Industrial Revolution.

Industry 4.0 symbolizes, allied with cloud computing and artificial intelligence, the tendency toward automation and data exchange in manufacturing technologies and processes which include IoT.

1.1.1 Maintenance Types

The whole scope of maintenance tasks is defined by the British and European Standard 13306 as "the combination of all technical, administrative and managerial actions during the life cycle of an item intended to retain it in or restore it to a state in which it can perform the required function" (BSI, 2010). The aim is clear: the equipment must be in a state in which it can perform its required function and, to guarantee this state, its components must perform flawlessly. If an equipment's component suffers from a premature malfunction or wears out, it could compromise the ability of the equipment to perform its function and thus may provoke a halt in production. Manufacturing companies desiring to keep equipment downtime at a minimum must have a proactive equipment maintenance policy – to reduce all implicit costs (Dhillon, 2006). In terms of maintenance strategies, two main categories can be adopted, namely:

1. Unplanned maintenance

   This is the simplest form, as nothing is planned forehanded.

   a. **Reactive Maintenance**

      Equipment is repaired after faults have occurred. If no failures occur, the maintenance is inexistent. If failures do occur, production halts, labour and spare components may be unavailable and downtime accumulates, which may result in costly maintenance.

2. **Planned Maintenance** (Deshmukh and Garg, 2006)

   This strategy presumes a thoroughly anticipated planning of maintenance tasks.

   a. **Preventive Maintenance** (PvM)

      Maintenance tasks occur in regular intervals and are scheduled based on working hours or cycles: all wear-prone components are checked and replaced.

      The occurrence of an unexpected failure is dramatically reduced and unpredicted downtime is less likely to occur. This type of maintenance is performed at regular

intervals, whether it is necessary or not, and the schedules are based on historic data using Mean Time Before Failure (MTBF). This means that maintenance downtime may occur without being necessary.

Additionally, there is no guarantee that all serviceable components are identified and replaced; This type of maintenance has the advantages of allowing the on-time ordering of spare components and reserving the necessary labour resources for the scheduled intervention.

b. **Predictive Maintenance** (PdM)

The enforcement of PdM comprises the observation of an equipment's state during its working cycle and take the decision of performing maintenance or servicing of the equipment based on indicators exposing abnormal or unusual working parameters.

In practical terms, the execution of PdM implies that working conditions and equipment's status are closely monitored to evaluate if the present conditions can lead to a potential failure. The equipment's components are monitored to reveal abnormal working conditions or unusual wear, and time to failure may be predicted.

The major advantage of PdM is that routine or regular scheduled maintenance can be anticipated or predicted, leading to the execution of maintenance tasks targeting a very specific or isolated component. These interventions avoid that a potential upcoming failure which would otherwise lead to a heavy breakdown and consequently long down-time of the equipment was detected and resolved anticipatedly. This is a step forward in maintenance, as some of the presumptions of scheduled maintenance are anticipated by carefully observing the equipment in their working environments.

To perform the predictions in PdM, data captured from the equipment's sensors must be processed by data analysis tools, such as Data Mining and Machine Learning (DM and ML) algorithms, to detect operational irregularities in equipment's working cycles so that they can be identified as forthcoming breakdowns.

In PdM, the maintenance is only performed if the monitored parts display signs of anomalous or abnormal behaviour or wear, which may lead to a near-future failure, meaning that downtime still occurs, but only if present equipment's status determines the need for intervention.

Highly efficient companies will put aside unplanned maintenance, opting in its place for planned maintenance. As discussed before, downtime (either unnecessary or forced) is to be avoided, which favours the adoption of PdM as the preferred approach. To enable this, data from the equipment must be collected, normalized and analysed. Internal or external sensors strategically positioned to monitor components to be evaluated in terms of possible failures can be used to capture this data at established time intervals (e.g. in seconds or milliseconds). Equipment work cycles generate physical indicators, such as heat, pressure and vibration: the

data captured is, therefore, a strong indicator of the working condition of the components the sensors are monitoring.

1.1.2 Sensor Types

Literature gives us an understanding of the most common types of sensors applied to equipment monitoring and data gathering and how these are applied to facilitate different wear indicators. Briefly, we consider:

Mechanical sensors are a class of sensors sensitive to changes in mechanical properties and physical parameters and convert them to an electrical signal (Sharma, 2019):

1. **Accelerometer Sensor:** detects the rate of motion of an object's velocity in a specific time interval, in Hertz (Hz);
2. **Humidity Sensor**: measures the percentage of water vapour in the air;
3. **Level Sensor**: detects levels of liquids in enclosed containers or tanks, in volume or percentage;
4. **Pressure Sensor**: senses the force applied to a surface per unit area, in kilograms per square centimetre ($kg/cm^{2)}$ or pound per square foot (lb/ft^2);
5. **Temperature Sensor**: measures the amount of dissipated heat, in degrees Celsius or Fahrenheit;

Electrical Sensors are electronic devices that sense current and voltage:

1. **Amperemeters**: measures the electrical current flowing in an electrical circuit, in Amperes;
2. **Voltmeters**: measures the voltage applied to an electrical circuit, in Volts;

The collected data, when compared to standard data of well-performing equipment components, should show if the observed deviations reveal decreasing performance and thus an indication of an approaching failure.

The purpose of this work focuses on the identification of data sources provided by equipment and manufacturing processes and their representation through means of ontologies and how these ontologies describe the temporal factors implied. The usage of ontologies to represent the concepts in the domain of industrial equipment to achieve a context representation and a context reasoning for PdM has been proposed in the literature (Giustozzi, Saunier, & Zanni-Merk, 2018) and further exploited in this work. Reasoning with ontologies has found applications for providing diagnostics and recognition of qualitative fault states for PdM purposes (Cho, May, & Kiritsis, 2019).

1.1.3 PiANISM Project

The PIANiSM project's goal is to facilitate the implementation of prescriptive and predictive maintenance approaches in the plastic extrusion industry, comprehending different domains and technologies to be applied to real-time data acquired from industrial equipment, sensors, and management software.

The data acquisition has as primary source a set of sensors located in the extrusion machines and as secondary source data acquired from the ERP management software, describing the work orders, materials, and manufacturing processes. All the data collected is in its nature time-sensitive and timestamped. Thus, sensor data is complemented with contextual information about the processes occurring at a particular time.

Ultimately, the goal of this process is to execute ML and DM algorithms to detect and predict abnormal working conditions, identifying eminent failure states and the calculation of the remaining time before the complete failure of specific components. Therefore, it is important to analyse, clean and format the data obtained via these heterogeneous sources, and to represent it through means of an ontology.

To adequately model the unstructured time-sensitive data for PdM purposes an ontology is proposed to cover the domain of the problem.

The proposed ontology must not only simplify the understanding of data collected from the different components of the system but, more importantly, allow the correlation of the previously unrelated data and the generating of new knowledge and insight, not possible to envision with the raw data supplied from the data sources.

1.2 Problem

As technology evolves, new approaches and resources are available to address previous out-of-reach solutions. The evolution of sensor technology and the range of available sensors having applicability to IoT right out of the box (Sharma, 2019) is a booster for technological change. Sensors used in the IoT world provide data ready to be digitally collected and analysed. Associating the features and capabilities of these sensors with the fact that mass production is a price-lowering factor, means that state of the art technology is available at a reduced cost. Increasing Cloud processing capacity has also contributed to this trend.

To be able to intervene before faults effectively take place, data regarding equipment status must be collected by the sensors capable to identify the equipment's points of failure. The data must be accurately seized by adequate sensors, (e.g. temperature, pressure, force, vibration, electrical voltage and current, et.al), but raw sensor data must undergo pre-processing actions before it can be used to generate insights about equipment behaviour. After it is properly transformed and cleaned, it can be processed in useful time, properly formatted, and

categorized, to have the necessary value to be delivered and interpreted by the algorithms deciding on the evaluation of potential faults.

The problem is the absence of mechanisms to predict faults on equipment operating in continuous work cycles, resulting possibly in a high level of maintenance requirements. Knowing that equipment faults cannot be avoided and are certain to occur because of the number of moving components prone to wear, extreme working conditions or workload may lead to accelerated degradation, but data gathered from sensors during operation can be analysed to understand patterns and predict future moments of malfunctions.

The main goal of PdM is to intervene before equipment faults effectively take place, avoiding increased downtime and maintenance costs. To achieve this, it is paramount to acquire real-time equipment status data, and process it in useful time.

One way to achieve this is by having the real-time acquired data, properly validated, sanitized, and formatted to be analysed and compared to standard reference data representing the acceptable tolerance in fluctuance over specific time periods. While incoming data is in the scope of the acceptance criteria of the tolerances, the equipment appears to operate in the desired working conditions.

Otherwise, data provided from one or multiple sensors, either in an isolated or combined pattern, that indicates variations compared to the acceptable tolerance levels and exceeding the tolerable time intervals, may be evidence of malfunction.

In this case, the company's maintenance software, either the Enterprise Resource Management (ERP), Manufacturing Resource Management (MPR), or the Computerized Maintenance Management System (CMMS) could receive a notification that specific equipment is exhibiting symptoms of potential component failure and potentially trigger preventive action.

To solve the problem of providing a reliable source of data than can be fed to DM and ML algorithms and simultaneously have the data semanticized in human readable format, as well as accounting for the temporal representation of the data, a mean to describe and relate the data must be provided beforehand.

Thus, the root of the problem to handle, resides in the characterization of the field of manufacturing processes and the representation of its time-sensitive data.

1.3 Objectives

The main objective is the development and application of an ontology to a real-world scenario, testing its performance and compliance to correctly evaluate previously collected and structured data from equipment sensors. The sensor data to be collected is supplied in real-time via a Web-Service. The overall goals of this work are detailed and explained next.

The main goal of this work is to develop a domain ontology that describes PIANiSM's application domain: PdM for plastic extrusion machines. Sensor and contextual data (e.g. supplied by the ERP) must not only be described semantically to enhance interoperability and application of ML and DM algorithms, but also accurately describe its temporal nature, as the data is continuously captured in near real-time and any predictions will also be time-dependent. Furthermore, the transformation and storage of the data obtained from those different sources into individuals of the proposed ontology is also covered in this document.

The proposed ontology's main goal is to achieve a semantic representation of the collected sensor data and manufacturing process data, having the capability to represent the semantic data temporally evidencing the time-sensitive nature of the data.

The added value of applying a process to semanticize the collected data is:

- to achieve a human readable format;
- to have a temporal representation of the data;
- to have the data correlated (machines, sensors, orders, processes, etc.);
- to have the data in a suitable representation for the application of ML and DM algorithms.

Once the data is semanticized in accordance with the application of the proposed ontology, assessments must be conducted to validate if the pretended correlation and representation is *de facto* achieved.

1.4 Approach and Development Process

This work will start with a literature review of existing ontologies, approaches for predictive maintenance, and existing semantic representations of time-sensitive data. This research will allow a better understanding of the benefits and drawbacks of the existing approaches to the problems this work wishes to address.

In terms of the process approach, a software engineering method to study contemporary phenomena in their natural context is to be employed (Runeson & Höst, 2019).

This method consists of five main steps:

1. case study design: which involves the definition of objectives and case study planning;
2. preparation for data collection: describes procedures and protocols for data collection;
3. collecting evidence: entailing the execution with data collected on the studied case;
4. analysis of collected data;
5. reporting.

This process ensures that the researcher is not limited to a strict laboratory environment. In opposition to a controlled environment, the researcher studies the phenomena in the natural context they occur, allowing for a holistic understanding of the subject's interaction with its context. Furthermore, the subjects and objects studied are not based on statistically representative samples. The research findings are obtained through the analysis of typical cases (Runeson & Höst, 2019).

Most of the work concerning the development of the purposed ontology will be done using the Protégé (Stanford Center for Biomedical Informatics Research, 2021) open-source ontology editor, as it is free to use and provides all the necessary tools to develop an ontology through means of a graphical interface. Furthermore, to demonstrate the application of this ontology, a software application developed in Java programming language will be developed, with the aim of transforming the data obtained through the different sources into individuals of the proposed ontology.

This work also involves the evaluation of distinct approaches of implementation, as these are meant to be compared to understand which will better fulfil the requirements of the project.

Finally, the completed work shall be documented in detail regarding the architectural design having into consideration software patterns and the ontology model developed.

1.5 Document Structure

The first chapter of this document, Introduction, defines and contextualises the problem and describes the main objectives of this project and the approaches employed to achieve them.

In the second chapter, the Background Knowledge, ontology concepts and principles are presented to contextualise the purposed solution.

In the third chapter, Technologies and Methodologies, the set of selected methodologies, tools and technologies are presented

In the fourth chapter, the State of the Art, a synthesis of the current conceptual and scientific approaches related to the problem in question is abridged.

In the fifth chapter, Value Analysis, the presentation of the New Concept Development (NCD), the Value Proposition, the Quality Function Deployment (QFD), are presented, and the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) evaluation is performed.

In the sixth chapter, Solution Design, a detailed solution for the problem to be solved with this dissertation is presented and explained in detail.

In the seventh chapter, the Ontology Design, the proposal and detailed design of a domain ontology bridging the gap between existing ontologies is made.

In the eighth chapter, the Ontology Implementation, describes the implementation of the domain ontology using Protégé. Detailed descriptions of the implemented classes, properties relations and attributes are presented.

In the ninth chapter, JSON to Ontology Mapping Tool, the detailed presentation of the architecture and the development of a software tool, making a functional application of the developed domain ontology.

The tenth chapter, Experiences and Evaluation, presents all the formulated research hypotheses and the methodologies to evaluate them.

Finally, in the eleventh chapter, the Conclusion and Future Work, as the name indicates, describes the conclusions of this dissertation and discusses potential future work.

2 Background Knowledge

In this chapter, a review of the background knowledge concerning ontologies is presented, namely ontology representation languages, ontology entities and properties, necessary to understand the objectives of the purposed solution.

2.1 Ontology

In this sub-section, the fundamental notions about ontologies and the Semantic Web will be addressed to provide a theoretical context of these technologies within the scope of the present problem.

2.1.1 Background

In philosophy the study of concepts of existence, "being" and "reality" are the domain of the ontologies. The term ontology itself has a Greek origin composed of the word "*onto-"* meaning 'the being', 'the thing' or simply 'that which is' and *-logia* meaning the logical discourse or logical speech. The origins of the study of the ontological subject can be traced back to Parmenides in his work entitled 'On Nature', where he first proposed a classification of the nature of existence with two distinct views, firstly that "nothing comes from nothing" and thus that "existence is eternal". This set the fundamental observation that existence is what can be conceived by thought, created, or possessed (Parmenides, 5th c. BCE). Overall, in philosophy, the study of ontologies addresses questions of how entities are classified into categories and the nature of these entities and how they exist at an essential level.

In the 1970s the possibility of developing algorithms with Artificial Intelligence (AI) began to emerge. This possibility identified the need for the formulation of knowledge as a basis for the development of these algorithms. Researchers in this area realized that they could create computational models that would allow the development of reasoning automatisms; however, it was only in the early 1980s that the scientific community began to use the term ontology to

refer to modelling theories and components of knowledge-based systems. It will be in the 1990s, with the article "Toward Principles for the Design of Ontologies Used for Knowledge Sharing" (Gruber, 1995), that the first definition of ontology appears as a specification of a conceptualization, being an ontology defined as a technical term designating an artefact that allows knowledge modelling of a given domain. Ontologies are then used to define specific content rules for sharing and reusing knowledge between software entities. An ontology is the description of concepts and relationships that exists formally between entities. The complementary definition indicates that an ontology is an explicit form of conceptualization, with the conceptualization being an abstract and simplified form of the world view that we intend to represent for a given purpose (R.Gruber, 1993).

The solution was to adapt the philosophical concept of ontology to a data model in the form of a graph that could support the definition of categories, properties, and relationships between entities. The application of this concept resulted in the computational representation of entities, enabling the construction and transmission of the knowledge of an entity to applications or computer algorithms. With this transmission of knowledge, it became possible to identify an entity in a context normally only accessible to human beings: the recognition of an entity based on the written transmission of human language. In other words, based on an ontology that describes an entity, it became possible for an algorithm to recognize the existence of the entity in a written text by identifying its characteristics.

An ontology, therefore, defines the concept or existence of a concept or object, based on an explicit specification of a conceptualization – it describes a specific domain or theme, called a domain of knowledge. The specification of a conceptualization means that an ontology is a description of the concepts and relationships of a given object and therefore maps entities or concepts in a text, in a set or data set. The ontology establishes the concepts and the relationships between them in an unambiguous way and in such a way that they can be processed by both human and machines.

In computer science, ontologies have as purpose the modelling of knowledge describing a domain, at a single or multiple threaded levels an ontology represents a set of concepts, the way they are related and their properties. This structured knowledge represents the infrastructure that will allow the knowledge to be extracted from ontologies to be provided to data search engines and inference engines. Therefore, they provide the knowledge that can be used to identify concepts, thus allowing the sharing of that knowledge between information systems.

2.1.2 Ontology Languages

The evolution of the ontology concept in philosophy to an ontology in computer science was made by using formal languages to represent concepts syntactically and semantically.

The fact that formal languages are used, ensures that a defined syntax guarantees the correct expression of a statement, and the application of semantic rules guarantees the consistency of the declarations.

2.1.2.1 RDF and RDFS

The RDF (Resource Description Framework) is an infrastructure that includes a data model, a set of syntactic rules, and a vocabulary. The RDF data model was published by the World Wide Web Consortium (W3C) in 2004 (w3.org, Resource Description Framework (RDF), 2004) and implements a relationship of entities or classes representing knowledge through a graph representing concepts semantically by nodes and the relations by the graph's edges. It aims to make descriptions of resources, namely WEB resources, and share the knowledge amongst different systems and algorithms.

RDF uses semantic triples based on the *"entity–attribute–value"* model to represent relationships between entities. A triple takes the form of *subject-predicate-object* and is the atomic data entity in RDF enabling the representation of knowledge in a machine-readable format. A triple is a set of three entities to codify semantic data, where the subject identifies the entity or resource, the predicate identifies a relationship or attribute that links the subject and the object, and the object represents the value attributed to the relation. An example of a triple can be observed in Figure 1.

Figure 1 - Triple Example

Syntactically in a triple the subject, predicate and object are identified and individually addressable by a Unique Resource Identifier (URI), by using the URIs the representation of an RDF graph is made possible.

The representation of an RDF graph to represent knowledge semantically is shown in Figure 2, to represent the German city of Leipzig and its mayor.

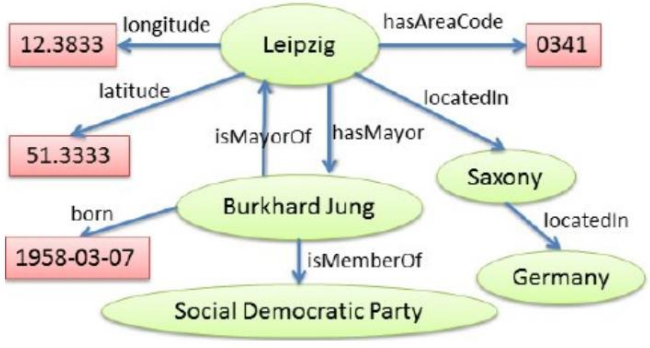


Figure 2 - Example of RDF graph representing triples (Auer, Lehmann, Ngomo, & Zaveri, 2011)

All the triples contained in the graph representing the German city of Leipzig and its mayor are detailed in Table 1.

Table 1 - List of Triples

Subject	Predicate	Object
Leipzig	longitude	12.3833
	latitude	51.3333
	hasAreaCode	0341
	locatedIn	Saxony
	hasMayor	Burkhard Jung
Burkhard Jung	isMayorOf	Leipzig
	born	1958-03-07
	isMemberOf	Social Democratic Party
Saxony	locatedIn	Germany

RDFS (RDF Schema) is an evolution and extension of RDF vocabulary by adding a schema allowing the taxonomic and hierarchical description of classes and properties. RDFS extends the definitions of RDF elements, enabling the definition of the domain, the range, and the hierarchization using vocabularies such as ***subClassOf*** and ***SubPropertyOf*** enabling the relation between the RDF classes and properties

2.1.2.2 OWL and OWL2

An evolution of RDFS was introduced by W3C in 2009 (w3.org, Web Ontology Language (OWL), 2009), the OWL (Ontology Web Language) extends the capabilities of RDFS by defining a language to process the data model and making deductions or inferences.

It presents itself as a language for describing ontologies with all the resources that can define much more complex relationships, together with restrictions on data values. In practical terms, OWL introduces descriptive and semantic capabilities to RDFS data models (Cambridge Semantics, 2021). The most recent version is OWL2 with the most recent edition published in 2012 and allows the possibility to define:

1. classes and Properties in the scope of a domain;
2. define individuals or instances based on the previously defined classes and properties;
3. enable the reasoning about these classes and instances.

In detail the OWL 2 Web Ontology Language (w3.org, OWL 2 Web Ontology Language, 2012) permits the definition of entities such as the ones represented in Table 2.

Table 2 - OWL2 Entities

Entity	Definition
Classes	Classes are used to group individuals that have something in common to refer to them, representing sets of individuals. Classes are used to classify a set of objects covered by a concept of human thinking E.g.: ***ClassAssertion(:City)*** represents the concept of a city, the ***ClassAssertion(:Person)*** represents the concept of all types of person.
SubClasses	To enable a system to take conclusions, a subclass axiom can be used to create special relationships in the form of hierarchies. E.g.: ***SubClassOf(:Woman :Person)*** declares that the class *Woman* is a subclass of the class *Person.* In this case, any individual of class *Woman* is implicitly an individual of class *Person*.
Individuals	Individuals represent actual objects from the domain. Named individuals are given an explicit name that can be used in any ontology to refer to the same object. Anonymous individuals do not have a global name and are thus local to the ontology they are contained in. E.g.: ***ClassAssertion(:Person :Peter)*** the individual *Peter* represents an individual of the class *Person*
Object Properties	Property to establish relationships between individuals, the properties can be described as attributes of an individual. E.g.: ***ObjectPropertyAssertion(:hasWife :John :Mary)*** represents an individual *John* that has as property *hasWife* representing the relationship with the value individual *Mary*
Data Properties	Property to establish relationships between individuals and data represented by a literal expression, the properties can be described as attributes of an individual. E.g.: **DataPropertyAssertion(:hasName :Peter "Peter Griffin")** represents an individual Peter that has as property hasName with the value “Peter Griffin”
SubProperties	SubProperty axioms are similar to SubClass axioms. A subProperty represents a hierarchical relationship between individuals. E.g.: ***SubObjectPropertyOf(:hasDog :hasPet)*** , states implicitly that the property *hasDog* is a subproperty of *hasPet*, meaning that all dogs are therefore also pets. Any individual with a relationship to the *hasDog* property implicitly has a relation to *hasPet.*

Data Types	Property that refers to sets of data values and allows them to be used in restrictions or to define a range. E.g.: ***DataPropertyRange(:hasAge xsd:integer)*** defines the set of integer values for the data property *hasAge*

2.1.2.3 OWL Sublanguages

The W3C Consortium published a range of OWL sublanguages presenting different capabilities and therefore may be used depending on the scope or needs of the ontology to be developed.

OWL Lite, has a set of basic constraints and allows the categorization of hierarchies:

- Some expressions part of the OWL Lite vocabulary are Class, Individual, sameAs, inverseOf;
- OWL Lite is limited in expressive power.

OWL DL, extends the former OWL Lite, having more expressivity by the inclusion of descriptive logic (DL) to enable reasoning:

- Some expressions part of the OWL DL vocabulary are oneOf, unionOf and hasValue, disjointWith, cardinality.

OWL FULL, extends OWL DL and delivers expressive capabilities and RDFS syntactic free expression. OWL FULL is the most expressive of OWL, it removes the remaining restrictions on OWL DL but becomes undecidable and impedes automatic reasoning:

- OWL Full and OWL DL use the same vocabulary, however, OWL has fever restrictions than OWL DL.

The expressive levels of OWL are represented in Figure 3.

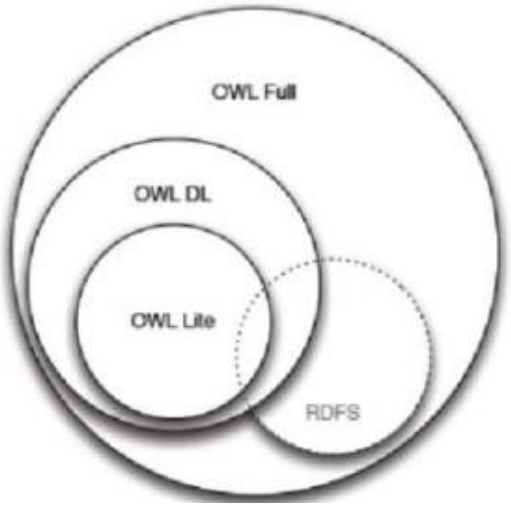


Figure 3 - OWL Sublanguages (Alamri & Bertok, 2012)

2.1.3 Inference

By formally describing a domain, not only through means of concepts and properties but also through restrictions, ontologies make it possible to conduct inference processes to generate deductions. The operation of creating a deduction based on an ontology is performed by algorithms called inference engines which, from the concepts and facts defined in the ontology, can extract additional, implicit knowledge based on the properties of the classes. This knowledge is presented in the form of consequences or logical deductions, to produce new knowledge without being explicitly stated. Inference is a process through which relationships and properties are used to materialize implicit knowledge.

An example of a deduction that an inference engine could make is demonstrated below.

In the knowledge base there are the following assertions:

Mary -> is a -> person.
A car -> is a -> vehicle.
A car -> is driven by -> Mary.

1. Knowing that the counter-domain of property “is driven by” is “a driver”.
2. Knowing that the inverse property of "is driven by" is "drives".

Assertions reachable through inference on the previous inference:

Mary -> drives -> a car.
Mary -> is a -> driver.

1. Knowing that the domain of the property "drives" is "a person".
2. Knowing that the domain of the property "drives" is "a driver".

Assertions reachable through inference on the previous inference:

A person -> drives -> a car.
A driver -> drives -> a car.

The new knowledge obtained from the inferences can be used for the formulation of new triples.

2.1.4 Language of choice

Considering the ontology languages background review, the language adopted in this work for the proposed ontology development will be OWL. This decision was made based on the semantic and interoperability characteristics of the language. The use of OWL allows the construction of complex relationships and the application of detailed constraints on concepts.

3 Technologies and Methodologies

In this chapter, methodologies, tools and APIs to fulfil the objectives of the purposed solution are described.

3.1 Methodologies

In this subsection, an overview of the adopted methodologies in the development process is provided.

3.1.1 Ontology Development 101

The Ontology Development 101 (OD 101) (Noy & McGuinness, 2001) methodology suggests an iterative ontology developing process.

This methodology emphasizes three fundamental rules to guide the design:

1. There is no perfect or correct way to model a domain. The most suitable solution shall be the one focusing on the concrete application;
2. The ontology development process should always be iterative, by firstly applying general concepts, and then iteratively, through reviews, refined and improved to a greater detail;
3. Concepts in the ontology should most likely be identified as physical or logical objects and should have a direct association with relationships.

In terms of the steps to follow when designing an ontology, the following sequence should be adopted:

1. determine the domain/scope of the ontology, and study the field of application;

2. study the possibility of reusing existing ontologies, incorporating them into the ontology to be developed;
3. elaborate a list of the most important concepts, these concepts shall originate the classes;
4. define the proper classes and the implicit hierarchy;
5. define Object and Data properties;
6. define restrictions and cardinalities;
7. create Instances or Individuals.

3.2 Tools and APIs

In this subsection, an overview of the adopted tools, programming languages and APIs used to implement the solution is provided.

3.2.1 Java SE and Apache NetBeans

As a programming language, to perform the implementation to fulfil the objectives, JAVA (Oracle, 2021) was paired with the Apache NetBeans Development Environment (The Apache Software Foundation, 2020). Java has the following advantages:

1. Apache NetBeans Development Environment:
    a. editor with integrated development tools;
    b. allows you to trace (debug) the application at run time;
    c. provides access to variables and the stack of accessed routines.
2. Multi-platform:
    a. windows, Mac OS and Linux.
3. Multi-environment:
    a. executes on a Virtual Machine in the Operating System;
    b. executes directly in an Internet Browser as an Applet.
4. Object-Oriented Programming:
    a. Classes, Inheritance, Polymorphism, Interfaces and Reflection.
5. Unit tests:
    a. integrated generation of template classes for unit tests.

3.2.2 Apache Jena

As a support framework, Apache Jena (The Apache Software Foundation, 2021) was chosen because it natively presents a range of functionalities to operate with ontologies. Apache Jena is a free-to-use open-source Semantic Web framework for use in conjunction with Oracle's Java SE. It provides an API to extract and persist data represented in graphs in RDF format, the graphs being represented as an "abstract" data model. These models can be accessed via data files, databases, URL's or a combination of those.

One of the advantages of Apache Jena is the native support for OWL (Web Ontology Language), also offering several inference engines. The functionalities used in this API focused on:

1. opening and reading an ontology;
2. extraction of classes and individuals;
3. extraction of class and individual properties;
4. creation of new ontologies (knowledge bases);
5. creating classes and individuals with their properties.

3.2.3 Apache Jena Fuseki

Apache Jena Fuseki (The Apache Software Foundation, 2021) is a triple store, capable of storing data in triples following the *subject-predicate-object* representation.

Apache Jena Fuseki acts as a Simple Protocol and Rdf Query Language (SPARQL) (Ontotext, 2022) server, being SPARQL a standard query language and protocol for Linked Open Data and RDF databases. SPARQL was designed to query a great variety of data, efficiently extracting information in non-uniform data and stored in various formats and sources.

Apache Jena Fuseki was installed as a Java web application combined with a UI for administration and executing queries.

The application provides two query and update protocols, the SPARQL 1.1 (W3C, 2013) and the SPARQL Graph Store protocol.

3.2.4 Protégé

Protégé (Stanford Center for Biomedical Informatics Research, 2021) is an open-source ontology editor that allows the creation and editing of ontologies and validation of the ontological model. The validations of the ontological models are performed through inference engines that test the consistency of the models and additionally allow the inference of new knowledge rules based on the analysis of the rules or properties of the entities that constitute the ontology. For this dissertation, Protégé was selected as the tool to be used for handling and maintenance of the domain ontologies to represent equipment, sensors and their relationships.

4 State of the Art

In this chapter, an overview of the state-of-the-art of the existing conceptual and scientific approaches in the field of Ontology-based PdM is performed. The methodology of research for conducting the state-of-the-art review is presented, focussing on the domain of PdM, Semantic and Time-Sensitive Data. From the result of the research, a set of works are introduced and presented. The state-of-the-art review describes existing resources, mainly already available ontologies in the field of PdM, Time-Sensitive Data, Sensors, and Extrusion Manufacturing processes.

4.1 State of the Art Review Methodology

To discover the existing works related to the targeted domain, a state-of-the-art review was performed. The research methodologies applied during this work aimed scientific journal articles and conference papers published in the English language between 2011 and 2021. The research was conducted using the following web search engines:

1. Science Direct[1];
2. Semantic Scholar[2];
3. Google Scholar[3].

Additionally, to limit the search results a filter was applied to narrow the search to the field of study of “Computer Science” and “Engineering”. A set of search keywords representing the targeted resources was established to serve as the base for the search queries and is summarized in Table 3.

[1] “Science Direct” [Online]. Available: https://www.sciencedirect.com [Accessed: 01-Fev-2021].

[2] “Semantic Scholar” [Online]. Available: https://www.semanticscholar.org [Accessed: -01-fev-2021].

[3] “Google Scholar” [Online]. Available: https://scholar.google.com [Accessed: 01-fev-2021].

Table 3 - Search Keywords

Domain	Industry 4.0	Intervals
Machine Learning	Modelling	Ontology
Patterns	Predictive Maintenance	Reasoning
Semantics	Temporal	

The search queries consisted of the conjugation of the keywords using the operator “AND”. The final queries used, and the results obtained using each of the Web Search engines are shown in Table 4.

Table 4 - Bibliography Search Results

Search Queries	**Semantic Scholar**	**Science Direct**	**Google Scholar**
"Industry 4.0” AND "ontology" AND "Semantic"	470	199	4570
"Industry 4.0" AND "ontology" AND "modelling" AND "Semantics"	258	1230	1470
"Ontology" AND "Reasoning" AND "Temporal" AND "Intervals"	590	782	318
"Predictive Maintenance" AND "ontology" AND "domain" AND "Machine Learning"	388	123	1030
"Temporal" AND "Ontology" AND "Patterns"	1420	147	704

The selected queries returned a very large number of results, too large to perform a thorough selection. The following steps were taken to condense the search results allowing the focus on the papers with the most pertinent context:

- from each applied search query, the top 10 hits from each search engine were considered;
- from the previous set of hits, the papers appearing in at least two search engines were considered;
- from the previous set of selected papers, the abstract of each paper was the subject of careful reading to evaluate and assess the focus on the intended domain.

Six papers resulted from this selection as they revealed themselves as those that best address the application of ontologies to support PdM and the modelling of ontologies to define time-sensitive qualities.

The selected papers and authors are presented in Table 5.

Table 5 - Selected Papers

Paper	Author(s)
Context Modeling for Industry 4.0: an Ontology-Based Proposal	(Giustozzi, Saunier, & Zanni-Merk, 2018)
Bridging the Gap Between Domain Ontologies for Predictive Maintenance with Machine Learning	(Canito, Corchado, & Marreiros, 2021)
Ontology patterns for the representation of quality changes of cells in time	(Burek, Scherf, & Herre, 2019)
Ontology-Based Representation and Reasoning about Precise and Imprecise Time Intervals	(GHORBEL, HAMDI, & METAIS, 2019)
A semantic-driven approach for Industry 4.0	(Cho, May, & Kiritsis, 2019)
CHRONOS: A Tool for Handling temporal Ontologies in Protégé	(Preventis, Marki, Petrakis, & Batsakis, 2011)

The selected papers are presented in the following sub-sections exposing the present state of the art in the domain of the proposed work.

4.2 Bridging the gap between domain ontologies for predictive maintenance with machine learning

The domain of PdM, already accounts for the existence of several ontologies for fault detection and diagnosis, for the characterization of industrial equipment and the descriptions of sensors, and ontologies describing manufacturing processes, each ontology models these domains independently without any direct relation (Canito, Corchado, & Marreiros, 2021). Moreover, none of the existing ontologies considers the temporal representation of time-sensitive data in a way that meets the requirements of PdM.

In the age of Industry 4.0 and IoT, data can be collected remotely from equipment and manufacturing processes. The acquired data is due to its nature, time-sensitive and thus representing changes in equipment behaviour over time. PdM must rely on the extraction of information from this data to apply the necessary evaluations and analyses to comprehend the variations over time of patterns in the data. Such analyses are commonly done through the application of ML and DM algorithms. As the existing ontologies fail to represent the time-sensitive dimension, not applying any temporal representation or time constraints, the effectiveness of the algorithms applied to the collected data is strongly reduced.

The present studied paper “Bridging the gap between domain ontologies for predictive maintenance with machine learning” (Canito, Corchado, & Marreiros, 2021), approaches the

existing gap in temporal representation and presents a group of existing ontologies that combined, propose a possible domain model for PdM, with the focus on the temporal representation of the data, an thus more suitable for the application of ML and DM. The already publicly available ontologies of interest, representative of re-usable concepts to be considered, are described in the following subsections.

4.2.1 CDM-Core Ontology

Name: CDM-Core ontology - CREMA Data Model - A Manufacturing Domain Ontology in OWL2 for Production and Maintenance

Purpose: CDM-Core is a publicly available ontology, for manufacturing process description, specialised in the application of sub-domains of exhaust car manufacturing and metallic press maintenance (Mazzola, Kapahnke, Vujic, & Klusch, 2016), including a set of classes defining component conditions and features of interest.

The SSN (Semantic Sensor Network) ontology is imported extending the domain to machine sensors.

namespace: http://www.example.org/CM/CM_global_ontology.owl#

url: https://sourceforge.net/projects/cdm-core/

4.2.2 ExtruOnt Ontology

Name: ExtruOnt Ontology for representing an Extruder Machine

Purpose: ExtruOnt is a publicly available ontology, describing a type of manufacturing machine, more precisely, a type that performs an extrusion process (extruder). Although the scope of the ontology is restricted to a concrete domain, it could be used as a model for the development of other ontologies for describing manufacturing machines in Industry 4.0 scenarios (Ramírez-Durán, Berges, & Illarramendi, 2019).

namespace: http://bdi.si.ehu.es/bdi/ontologies/ExtruOnt/ExtruOnt

url: http://siul02.si.ehu.es/bdi/ontologies/ExtruOnt/docs/

4.2.3 SSN Ontology

Name: Semantic Sensor Network Ontology

Purpose: SSN is a publicly available ontology, describing sensors and observations, and related concepts. The focus is the description of sensors and their observations, the involved procedures, the studied features of interest, the samples used to do so, and the observed

properties, as well as actuators (W3C, 2011). However, the ontology does not describe the sensor's domain concepts, time relations or locations.

This ontology describes how sensors observe properties, and how these properties are translated into symptoms, fault states, component conditions and component faults. The concepts provided by this ontology can be extended to cover PdM purposes and analysed by any type of ML and DM algorithms.

Equipment sensors generate streams of unstructured time-stamped data, lacking formalized structured representation, leading to weak interoperability among different systems and low re-usability. The Sensor Ontology's goal is to perform the annotation of unstructured data with formalized semantics, meaning that by creating semantic context and contextual information, the data's usage and interoperability are increased. This is the direct applicability for sensor data to be applied to PdM.

The conceptualization of sensor measurements related to sensing and observations is reused from the SSN ontology. It comprehends several datatype properties representing sensor features, such as ***observes*** (relation between a sensor and a property that it is capable of sensing) and ***resultTime*** (instant of time when the observation was completed).

namespace: http://purl.oclc.org/NET/ssnx/ssn

url: https://www.w3.org/2005/Incubator/ssn/ssnx/ssn

4.2.4 Time Ontology

Name: OWL-Time

Purpose: OWL-Time is a publicly available ontology of temporal concepts, describing the temporal properties of resources in the world. The ontology provides a vocabulary for expressing facts about instants and intervals, together with information about durations, and temporal position including date-time information (OGC & W3C, 2020). This ontology has a solid representation of time constraints, and an extended set of properties that can be used to represent overlaps and sequences.

Time or time intervals are fundamental information to manage PdM efficiently, as senor data varies as time advances. The Time Ontology comprises all information related to the current time and provides timestamps for all context information that may change over time. This ontology delivers vocabulary to describe complex interval-based temporal information

namespace: http://www.w3.org/2006/time#2016

url: https://www.w3.org/TR/owl-time/

4.2.5 Onto-DM Ontology

Name: Ontology of Core Data Mining Entities

Purpose: OntoDM is a publicly available generic ontology for the domain of data mining. The ontology includes the information processes that occur in the domain of data mining, the participants in the processes and their specifications (Panov & Stefan, 2020). This is a domain-level ontology, separating the data mining domain into three subcategories, by importing the following ontologies:

- Data types Ontology (Onto-DT);
- Core Data Mining Entities (Onto-DM core);
- Data Mining Knowledge Discovery (Onto-DM-KDD).

The core ontology is responsible for the representation of entities, like data and data mining tasks, the algorithms they utilize, and setting the boundaries of constraints to be applied to data, algorithms and data mining scenarios. This ontology represents several algorithm executions, such as Predictive Modelling Algorithm Execution and Pattern Discovery Algorithm Execution, demonstrating a starting point to analyse data from the concepts described in the formerly presented ontologies.

namespace: http://kt.ijs.si/panovp/OntoDM.owl

url: https://bioportal.bioontology.org/ontologies/ONTODM-CORE

4.2.6 Summary

The paper analysed presented an overview of presently existing ontologies suitable to be used in the domain of PdM applied to extrusion equipment, covering the domains of ML and DM, time-based constraints, the description of manufacturing processes and extrusion machines, not omitting the field of sensors necessary to describe the captured data.

4.3 Context Modelling for Industry 4.0: an Ontology-Based Approach

At the International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, work addressing the usage of Ontologies to represent industrial equipment resources, sensors, temporal relations and processes was presented (Giustozzi, Saunier, & Zanni-Merk, 2018). This approach describes the usage of ontologies to represent the concepts in the domain of industrial equipment, to achieve context representation and context reasoning. In this scope context is understood as implicit or explicit information used to characterize the situation of an entity, comprehending sensor inputs and service description.

The work analysed in this section exposes the context where PdM is to be applied based on ontologies. This section presents a condensed overview of the concepts considered the most relevant to be identified as useful in the field of PdM. This approach is based on the definition of context types to characterize the situation of entities. The contexts identified are, the entity, the location, time, and activity, allowing to answer the questions of who, what, when or where to deduce contextual information.

The main ontologies of the Context Ontology are presented in Figure 4. These ontologies can be classified as general, core, domain, and application ontologies, based on the conceptualization they represent. Core ontologies are domain-independent conceptualizations that can be reused across various domains of knowledge, the Time ontology, the Location Ontology and the Sensor Ontology are of this category.

Domain ontologies model concepts for certain domains, which apply to more specific scenarios, the Process Ontology, the Resource Ontology and the Situation ontology are of this category, their concepts describe industrial processes.

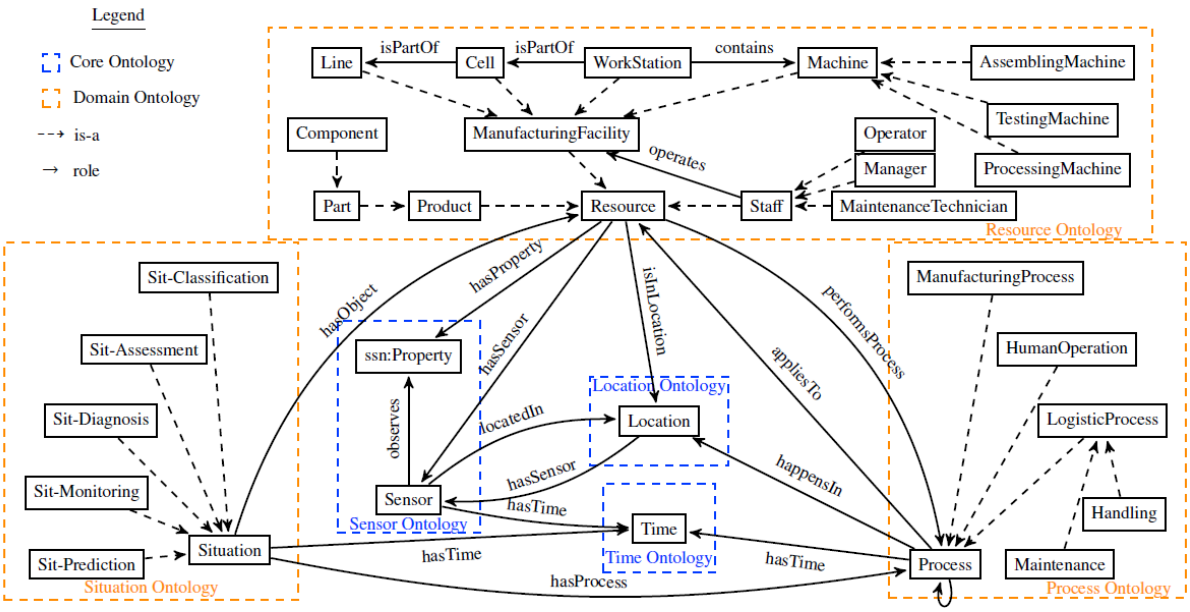


Figure 4 – The main concepts of the Context Ontology (Giustozzi, Saunier, & Zanni-Merk, 2018)

To understand how each of the ontologies composing the Context Ontology can be reused for PdM purposes, each of these ontologies is described in the next subsections.

4.3.1 Resource Ontology

The resource ontology describes human entities and physical objects, in the case of PdM, the latter ones are those to focus on. The physical objects are described by concepts representing devices, hardware and equipment, containing a collection of equipment categories used in

manufacturing processes. The resource ontology describes in detail the composition of a product line, machines or equipment, Workstations as groups of machines, Cells as groups of Workstations and Lines as groups of Cells. An application ontology for each type of machine, equipment or component is modelled, containing information about the machine's category, the operations it can perform and the overall characteristics. This information is modelled by each ontology, by instantiating concepts (classes) supplied by the domain level ontology, the Resource Ontology, assigning concrete instances (individuals) and values to their properties.

Particularly useful for the application of PdM is the *hasSensor* property present in individuals, which is used to assert that one or more sensors are attached to equipment to measure certain properties. In the case of sensors, the physical phenomena they target, are indicated by the ***hasProperty*** property. This description is one of the requirements for PdM.

4.3.2 Location Ontology

To be able to represent spatial locations and relations, the Location Ontology permits this representation of physical abstractions e.g.: a machine is located in a particular sector. The Location Ontology is aligned with the Spatial Relations Ontology.[4]

This ontology is linked with the SSN ontology through various properties:

- property ***hasSensor*** to express that a location has a sensor;
- property ***locatedIn*** indicates the location a sensor is in;
- property ***isInLocation*** indicates the location of a Resource;
- property ***happensIn*** identifies the location where a process takes place.

4.3.3 Process Ontology

In manufacturing, a Process represents a set of tasks performed in a specific context, e.g. occurring time, place and related resources. The domain ontology named Process ontology represents a taxonomy of processes existing in manufacturing. operations include machining operations classified according to their physical features.

The resource which performs the process is defined by using the ***performsProcess*** property. The location where a process occurs is represented by the ***happensIn*** property.

4.3.4 Situation Ontology

A situation describes the combination of several resources, located in several locations, or sensor measurements linked through spatial and temporal relationships. Situations are based

[4]https://data.ordnancesurvey.co.uk/ontology/spatialrelations/

on the type of task being executed by the equipment. The taxonomy adopted for the Situation Ontology expresses that, situations can be described as being of the following types: classification, assessment, diagnosis, monitoring and prediction (Schreiber, et al., 2000).

If the task performed is monitoring then the aim is to detect when the equipment behaviour is not normal during the execution, a situation, in this case, is a specific scenario identifying a combination of values in attributes provided by the sensors. If the combination of values reveals as not in the scope of the expected boundaries, hence this means that the situation could lead to a failure.

4.3.5 Time Ontology

This work also considers the Time Ontology, already focussed on the previously described paper "Bridging the gap between domain ontologies for predictive maintenance with machine learning" (Canito, Corchado, & Marreiros, 2021)

4.3.6 Sensor Ontology

This work also considers the Sensor Ontology, already focussed on the previously described paper "Bridging the gap between domain ontologies for predictive maintenance with machine learning" (Canito, Corchado, & Marreiros, 2021)

4.3.7 Summary

The work analysed demonstrated that it is feasible to have an ontology-based approach applied to a manufacturing environment. The approach is achieved by combining core ontologies (Sensor, Location, Time) with domain ontologies (Process, Resource, Situation) into a context ontology applied to a specific manufacturing environment. These ontologies provide a detailed description of the concepts and the attributes needed to have a solid base for PdM. Having presented the purpose of this dissertation, being able to ensure an ontology-based approach to PdM, and therefore the need to describe the equipment or resources, the sensors, the time intervals, the processes performed, the situation where anomalous readings are identified, the evaluated ontologies cover a large set of concepts to be considered.

4.4 Ontology patterns for the representation of quality changes of cells in time

An extensive study realized to represent quality changes of cells in time using ontologies (Burek, Scherf, & Herre, 2019) identifies a set of questions that also apply to changes in time-sensitive data in industrial equipment measured by sensors. The requirements specify the use of an ontology, for the representation of temporal information. Cell migration's patterns comprehend the changes of properties over time, to represent these changes, two patterns of temporally information changes are discussed: N-Ary relation reification and 4d fluents. Both schemes are formalized within the ontology language OWL. This study demonstrates that reification and 4d fluents are adequate and can be combined to design a Cell Tracking Ontology (CTO) for the purpose.

4.4.1 Tracking of changes over time

The challenges involved in tracking, from just a few cells up to tens of thousands of cells, over a time span from a few hours, up to several days, may represent a total number of individual observations from between mere tens of thousands to a few million. Each individual has a set of qualities changing over time, such as the Position, the Shape and the Dimension.

The observation of changes over time are formulated according to the following assumptions:

1. entities are designated as *Objects*;
2. objects have characteristics designated as *Qualities*;
3. a Quality can be predicated upon an Object, assigning that an object has *Quality Assignments*;
4. a Quality Assignment can change over time, the quality of an object can be different at distinct moments designated as *Time Entities*.

Summarizing the assumptions enables the formalization of: "an Object has a Quality at a specific Time Entity", this representation can be made using the pattern shown in Figure 5 (***Pattern A***), using OWL properties. The *OWL:class Object* linked with class *Quality* linked by OWL properties ***has_quality_at_[t].***

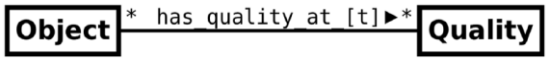


Figure 5 – **Pattern A** -Quality assignment modelled as time-indexed OWL property (Burek, Scherf, & Herre, 2019)

This approach is only efficient with limited time-indexes (**T-Index**). When the number of observations or number of Time Entities is very large, it would generate an unmaintainable number of quality assignments. Quality changes to be represented over time have strong

similarities with sensor data changes over time, as sensor data changes over time in terms of type, state, amplitude, et.al.

4.4.2 Patterns for modelling qualities

In the following sub-sections two patterns frequently proposed for modelling temporal information are presented, the reification of n-ary relations and 4d fluents.

4.4.2.1 N-Ary-Relation Reification

Reification (treating something immaterial as a material thing) is a frequently used strategy for modelling temporally changing information. It represents a time-indexed quality as a 3-ary relationship, linking an Object, its Quality and the Time Entity at which the Quality is assigned.

The reification strategy can be represented in the pattern shown in Figure 6 (***Pattern B***). In this case, the quality assignment is modelled as an *OWL:class Quality Assignment*, acting as a proxy (interface) between an object and its quality.

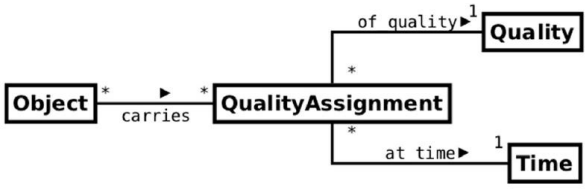


Figure 6 – **Pattern B** - Quality assignment modelled as time-indexed OWL class (Burek, Scherf, & Herre, 2019)

A reified Quality Assignment represents an assignment of a Quality to an Object and is therefore dependent on both the Object and the Quality. In this pattern, the time-indexed quality value assignments are represented as instances only. Even in situations where many time-indexed value assignments occur, the number of classes and properties in the ontology remains constant. However, the model introduces additional OWL classes and OWL properties for representing time-indexed quality attributions.

4.4.2.2 4d fluents

The 4d fluents pattern is based on a philosophical theory explaining the persistence of objects through time. The 4d fluent pattern is shown in Figure 7 (***Pattern C***). Compared with the reification pattern, 4d fluents don't reify a temporally indexed relation, but instead a temporal part of an object. In this paradigm, time-extended objects are sliced into temporal parts. The structural difference relative to the reification pattern is the cardinality of determining the number of qualities linked to a reified class, in this case *1..n*.

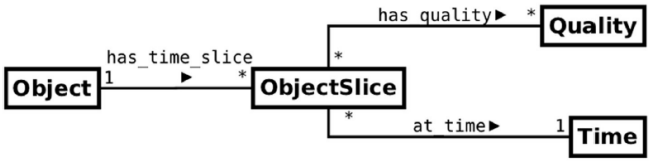


Figure 7 – **Pattern C** - Reified 4d fluents (Burek, Scherf, & Herre, 2019)

The cases when numerous quality assignments are present is the real challenge of ontology engineering. Therefore, the patterns must be analysed considering three different cases of the temporal distribution of qualities:

1. **Temporally non-overlapping quality assignments:** an object can have a quality at one time and another quality at another, but it can never have both qualities at the same time. This is the simplest case when an object has a single quality at a given time. Both previous described Patterns (B & C) are adequate;
2. **Temporally equal quality assignments:** at a single time point numerous distinct qualities are observed. This case is applicable when two or more quality assignments overlap temporarily. When for an object, at a given single time point more than one quality is observed, for each separate quality assignment a new instance has to be introduced. An alternative approach introduces temporal particles located at discrete time points, called *presentials*. In this way, an object observed at a single time point could be considered a presential object. A presential can have multiple assigned qualities present at the same time point. This enhanced pattern is represented in Figure 8 ***(Pattern D)***, modelling time slices and *presentials*.

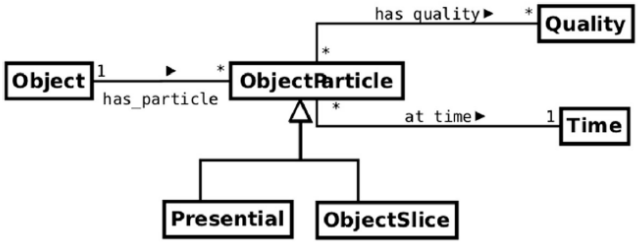


Figure 8 – **Pattern D** - Generalized 4d fluents. Presentials and Slices. (Burek, Scherf, & Herre, 2019)

The presential pattern reduces the number of instances introduced to the model, all coinciding quality assignments are modelled using a single presential instance. A slice can link an object with multiple qualities when quality assignments overlap temporally;

3. **Temporally overlapping, but not temporally equal quality assignments:** qualities change independently from one another. This particular case occurs, when an object has the same quality assignment over a sequence of time points t_1, t_2, ….t_n, deducing that the quality remains unchanged during the whole interval (t_1,t_n). Now, considering

that many independent quality assignments may temporally overlap, an object may have a constant *Quality A* during a time interval (t_1,t_5), but *Quality B* changes between interval (t_1,t_3) and interval (t_3,t_5). This raises the question of what the temporal part of an object is, and what rules apply to the slicing of an object into temporal parts. When all qualities assigned to an object in the same time span of a slice are attributed to the slice directly, this is ***vertical slicing***. This implies that each slice contains a full specification of the object at a given time.

This limitation can be overridden by an alternative approach, instead of slicing an entity vertically along the time dimension, ***horizontal slicing*** could be applied, slicing the entity along its quality assignments. This enables that a slice doesn't fully represent an object at a given time point, but only some of its qualities. Where a slice represents an indexed reified attribute of an entity.

4.4.3 Summary

The previous analysis shows that there are several approaches to model time-sensitive quality changes in ontologies. The performance of the previously presented patterns is evaluated by the simulation of changes of four parallel but distinct qualities (K,L,M,N) over a period t_1 to t_5 is shown in Table 6.

Table 6 - Number of elements for a fragment of ontology representing the change of four qualities of a single cell (Burek, Scherf, & Herre, 2019)

	Classes	**Object properties**	**Individuals**	**Object Property Expressions**
T-Indexed OWL Property	5	7	11	7
4d Fluent-Vertical	7	3	21	30
4d Fluent-Horizontal	7	3	25	24
Reification Pattern	7	3	28	30

Qualities K and L change independently from all others, and qualities M and N, change simultaneously. Overall the changes are distributed over 5 different moments in time [t_1,t_5], representing several changes in qualities (K,L,M,N).

The changes in qualities are shown in Figure 9.

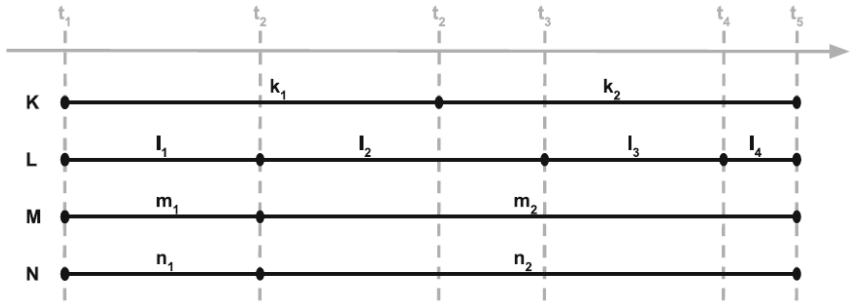


Figure 9 - A fragment of simulated cell tracking experiment results presenting changes of qualities K, L, M, N over time t_1 to t_5 (Burek, Scherf, & Herre, 2019)

There are several characteristics to be considered when choosing a pattern that is adequate for the ontology model of time-sensitive data, these characteristics are summarized below.

1. the **Time-indexed property** pattern is appropriate for cases with a limited number of time indexes. The number of object properties is proportional to the t-indexes;
2. the **Vertical 4d fluent** pattern is appropriate for cases with many time indexes and with temporally equal quality assignments. The number of object properties equals the number of quality changes, and the number of individuals created is proportional to the number of t-indexes and object properties;
3. the **Horizontal 4d fluent** and the **Reification** patterns are appropriate for the cases with overlapping but not equal quality assignments. The number of object properties equals the number of quality changes, and the number of individuals created varies depending on the temporally overlapping qualities.

The observed results lead to the conclusion, that the more complex the method of temporal reification, the more properties and individuals are created, but in the other hand there is a significant gain in terms of representativeness.

The changes in object qualities over time and the patterns analysed are applicable to the design of an ontology model to collect time-sensitive data from sensors. Sensor data changes at time periods *(t1, tn),* representing quality changes in data.

The analysed patterns provide a guide to be followed in the design of an ontology model applicable to PdM.

4.5 Ontology-Based Representation and Reasoning about Precise and Imprecise Time Intervals

The representation of time-sensitive data in ontologies, either comprised of precise or imprecise time intervals, poses a new question, how to properly implement the representation of changes in data over time? OWL or RDF can be used to represent temporal data not modifying OWL or RDF syntax. The temporal representation in RDF uses the annotation of properties with the data about the time interval they hold on, only using RDF triples. Several approaches have been proposed, Reification, Versioning, N-Ary Relations, 4d fluents and Named Graphs.

Of these approaches, different degrees of adequateness may be inferred. Reification suffers from redundancy, as new objects are created to represent temporal relations; Versioning handles the changes in time in ontologies by creating new versions of the ontology, requiring intensive searches in all ontology versions; N-Ary Relations represent an N-Ary relation as two properties, each related to a new object for every interval, however suffering from data redundancy; 4d fluents uses a 4th dimension being the temporal data to represent concepts varying in time as 4-dimensional objects, data redundancy is minimized as changes occur only in temporal parts; Named Graphs to represent time intervals as named graphs, where triples belonging to the same graph have the same interval or period, but this approach is not part of OWL.

Recalling all the different approaches, the one best suited to represent time intervals and qualitative time changes between them are 4d fluents, having as main concern the minimization of data redundancy. This approach is well suited to extend already existing OWL concepts, implementing data changes over precise or imprecise time intervals (GHORBEL, HAMDI, & METAIS, 2019).

4.6 A semantic-driven approach for Industry 4.0

In the scope of Industry 4.0, pursuing an architecture to join different sources of information into a single architecture capable of creating an information flow from sensors connected to manufacturing equipment, storing the collected data to be processed and classified enabling the data to be transformed into semantic data for PdM purposes, is proposed in “A semantic-driven approach for Industry 4.0” by (Cho, May, & Kiritsis, 2019).

Reasoning with ontologies has found applications for providing diagnostics and recognition of qualitative fault states for PdM purposes, using ontologies as a reference data model representing knowledge and enabling an efficient integration and management of data. Ontologies can therefore provide a meta-model, independently of the source data formats, that is understandable by machines and humans.

The transformation of data from multiple sources into semantic structured data is a complex task and no industry standards are defined. The inexistence of standards creates a gap in methodologies to synchronize and integrate industrial data with a semantic model.

A proposed architecture is represented in Figure 10 and consists of two vertical layers handling Knowledge Management and Analytics & Synchronization.

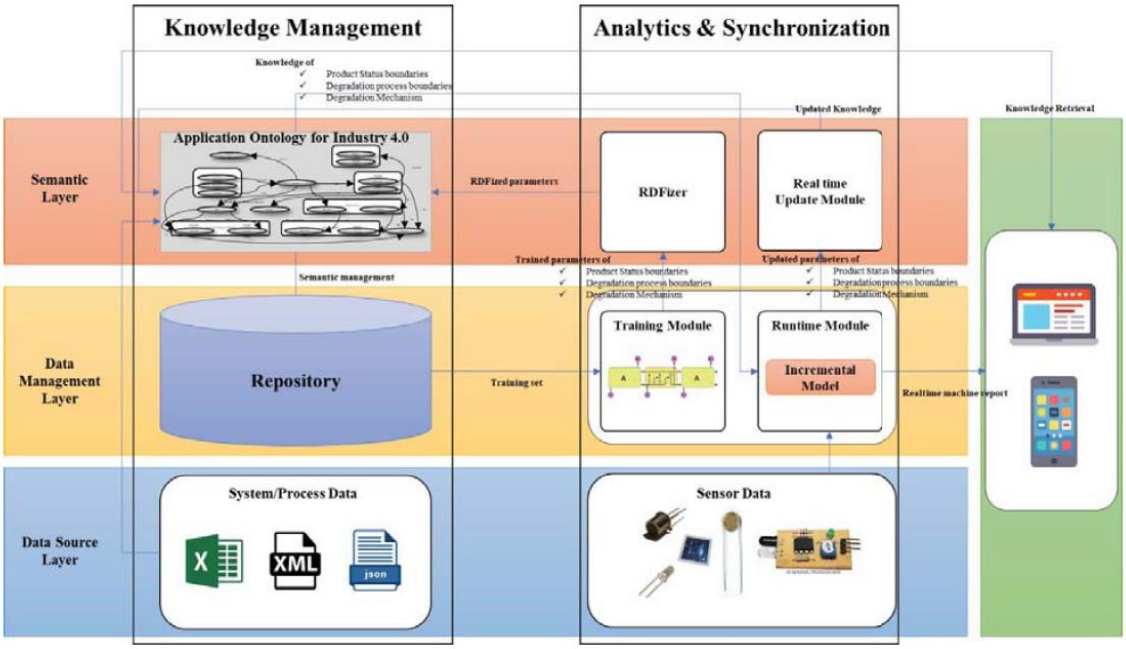


Figure 10 - Semantic Driven Architecture (Cho, May, & Kiritsis, 2019)

Having each of these layers, three sub-layers, a Semantic Layer a Data Management Layer and a Data Source Layer. For the present case, the first vertical layer provides integration of massive data via semantic technologies, where the Semantic layer is in charge of offering an ontology providing a data reference model with rules and data in-stances; the Data Management Layer embodies the repository that stores all the data in an RDF triple store, facilitation the retrieval of data based on semantic technology; the Data Source Layer representing all non-real-time data required describing the system and process data (Cho, May, & Kiritsis, 2019).

4.7 CHRONOS: A Tool for Handling temporal Ontologies in Protégé

The purpose of presenting the CHRONOS (Preventis, Marki, Petrakis, & Batsakis, 2011) tool is to expose how previous works undertook the challenges of representing temporal relations in ontologies. There is no intention to use this tool in this work, nonetheless, the representation of temporal relations has already been the subject of several works, and the adopted solution follows the path of 4d fluents or N-Ary relations. This challenge arises from the need to

accurately represent changes in classes and properties over time, to achieve a truthful representation of temporal entities in ontologies.

Knowing that temporal properties may have different values at different time points, the temporal representation often results in a complex and hardly readable ontology compared to a non-temporal ontology. This complexity is the result of the addition of additional classes to represent properties over time, while still preserving the original semantics, represented as temporal relations. This complex task was addressed and resolved with CHRONOS (Preventis, Marki, Petrakis, & Batsakis, 2011), a Protégé (Stanford Center for Biomedical Informatics Research, 2021) plug-in, which enables the manipulation of temporal relations based on Protégé's already existing methods to create Classes, Individuals, Data and Object properties.

An example of such a transformation can be observed in Figure 11 where the triple (subject-predicate-object) *John-**worksFor**-Company* was transformed into a temporal relationship by adding the *Event* individual to link the subject and the object. The event individual is related to a *TimeInterval* individual with the *during* object property. The *TimeInterval* individual, by its side, relates to two *TimeInstant* individuals representing the *Start* and *End* of the temporal interval.

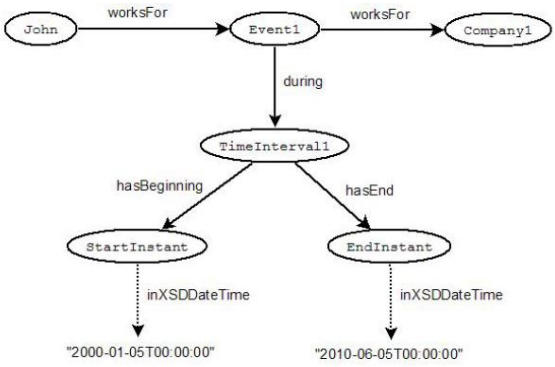


Figure 11 - Temporal object property between entities (Preventis, Marki, Petrakis, & Batsakis, 2011)

The described process facilitates the representation of static data, introducing a temporal dimension, and the resulting dynamic entities with N-ary relationships, follow the W3C recommendation for time representation. As observed, the resulting temporal representations of the initial static triple *John-worksFor-Company* present a simple and easily readable representation without the addition of many entities.

4.8 An Ontology to Promote Interoperability between Cyber-physical Security Systems in Critical Infrastructures

In the scope of PdM, when an anomalous situation is detected, and a fault state has identified the event in which the detection has occurred, should forward an alert to the systems capable

of taking actions based on the nature of the fault. A similar scenario is described in the paper "An Ontology to Promote Interoperability between Cyber-physical Security Systems in Critical Infrastructures" (Aleid & Canito, 2020), which aims the development of an ontology applicable to airport cyber-security.

This ontology accounts for the observation of events occurring in a specific moment and depending on the nature of the event an alert is triggered. Although the application of the described ontology was applied to a use-case of distinct nature than the application of PdM, the concepts utilized can be reused and applied to the domain of PdM. The base concept definition is shown in Table 7.

Table 7 - Concept Definition (Aleid & Canito, 2020)

Concept	Definition
Alert	A notification that a specific attack has been directed at an organization's information systems
Asset	Asset Information or resources which have value to an organization or person.
Event	A discrete change of state or status of an Asset or group of Assets. Specific Events may trigger Alerts
Incident	An Event (or group of Events) that compromises an Asset. An Incident may be retroactively classified as an attack. Additionally, it has some sort of impact within the organization, which is described by its severity and completion level.

The relation of the base concepts states that an Event can affect Assets and trigger Alerts. The severity of an Alert is related to the Criticality of the affected Asset. An Alert has several subclasses to comply with the type of alarm to trigger, such as an Alarm, a Warning, an Advisory or a mere Info. An Incident may be related to Events that occurred during a defined period.

The proposed cybersecurity ontology also extends already existing ontologies such as:

- UCO a Cybersecurity ontology (Syed, Pädia, Mathews, & Joshi, 2016)
- IODEF an Incident Object Description Exchange Format ontology (Danyliw, 2007)

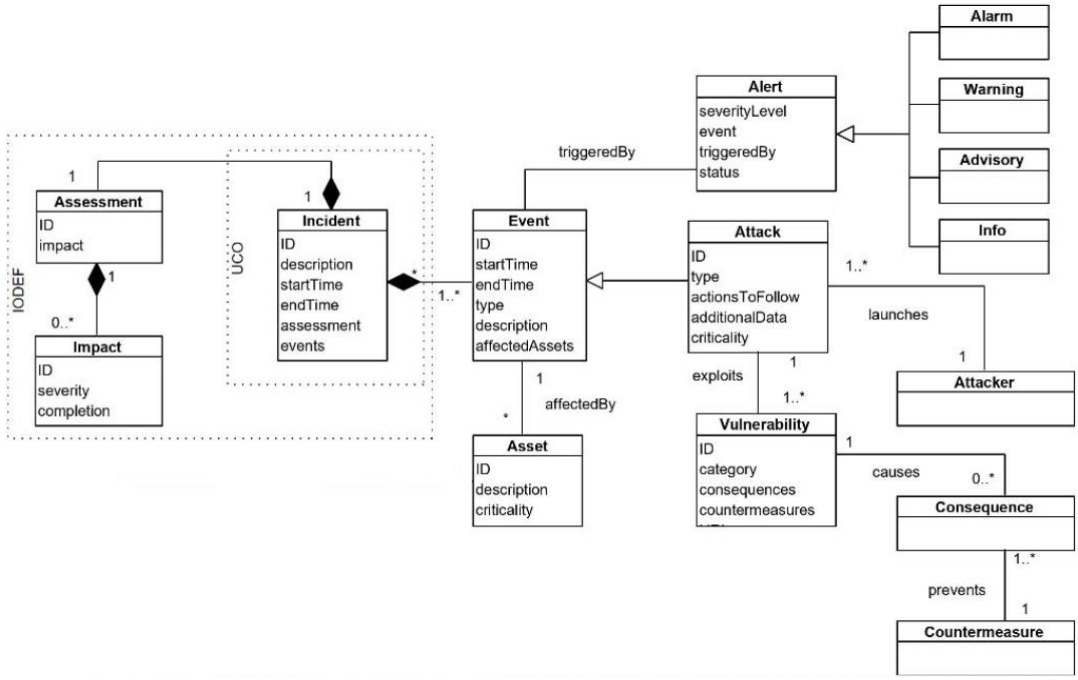


Figure 12 - Relationship between Incident, Event, Alert, Asset, and other concepts. (Aleid & Canito, 2020)

At the time of the publication of the work, the development of the proposed ontology was at an early stage not having even an adopted name.

Following the publication of the paper an ontology for Airport Security was developed in the Master Thesis “Development of an Integrated Ontology to Enhance Interoperability for Airports’ Security Solutions” (Aleid, 2020).

The developed ontology was named “Airport Security Interoperability Integrated Ontology (ASIIO)” and complemented with a more extended set of concepts.

The ASSIO ontology main concepts are presented in Figure 13.

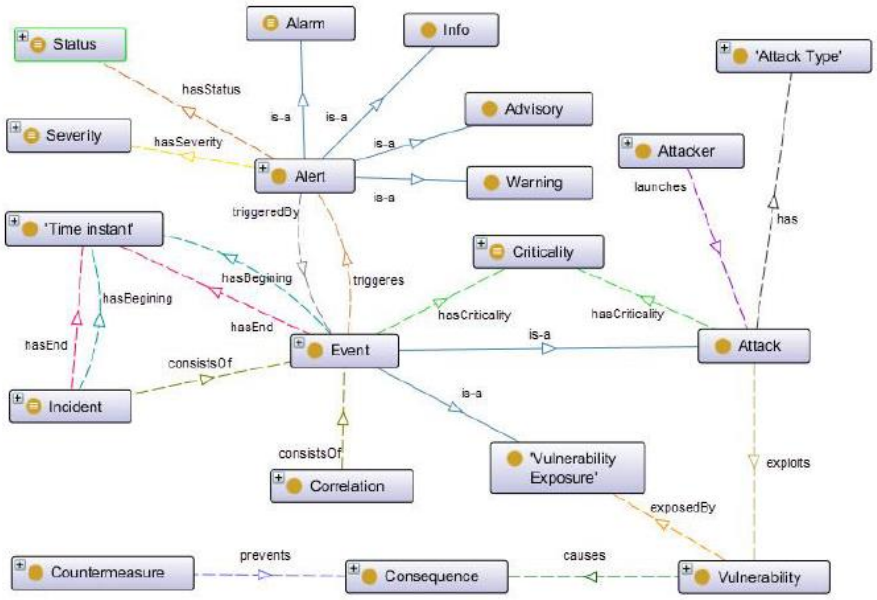


Figure 13 - ASIIO Main Concepts (Aleid, Development of an Integrated Ontology to Enhance Interoperability for Airports' Security Solutions, 2020)

The fully developed ASSIO ontology contains the necessary concepts to describe the alerts to be triggered in consequence of an incident during an event and thus reveals a high degree of applicability to the domain of PdM

4.8.1 ASIIO Ontology

Name: ASIIO- Airport Security Interoperability Integrated Ontology

Purpose: The Airport Security Interoperability Integrated Ontology (ASIIO) is an integrated ontology that combines cyber and physical security and was developed in the scope of the Security of Air Transport Infrastructure of Europe (SATIE) project (Aleid & Canito, Airport Security Interoperability Integrated Ontology, 2020).

namespace: http://www.gecad.isep.ipp.pt/ASIIO#

url: https://www.gecad.isep.ipp.pt/ASIIO_SITE/index-en.html

5 Value Analysis

Equipment maintenance is a critical factor in the manufacturing process. Traditional approaches such as reactive maintenance, only acting after a failure has occurred, and even Preventive Maintenance (PvM), acting before any failure has even occurred, have led to the emergence of a third way of performing maintenance. Not after the occurrence nor before any sustainable indicator of a fault occurrence is perceivable, predictive Maintenance (PdM) aims at only performing equipment maintenance when effectively concrete data attests to the working status of equipment components susceptible to failure in a predictable time span. PdM acts with anticipation of the occurrence, having the certainty that equipment downtime was not wasted and that the fault was identified at an early stage avoiding costlier repairs.

In this chapter, the value analysis of PdM is to be presented determining the most important characteristics of the proposal and business value.

This Value Analysis comprehends four evaluation methods, covering the development process, the customer value analysis, the assessment of customer demands and finally the use of a decision tool to determine the ideal solution to adopt.

1. **New Concept Development** (NCD) model for market-driven product development with Opportunity Identification;
2. **Value Proposition Canvas** (VPC) is used to evaluate that service is placed in the scope of the customer's values and needs;
3. **Quality Function Development** (QFD) is a process for designing a service based on customer demands, and assessing the engineering solutions to achieve the accomplishment of the service;
4. **Technique for Order Preference by Similarity to Ideal Solution** (TOPSIS) is a multi-criteria decision analysis method that compares a set of alternatives by identifying weights for each criterion.

5.1 New Concept Development

To establish a common process to describe the Front End of Innovation (FEI), and to improve the overall innovation process, New Concept Development (NCD) (Koen, et al., 2001) model was defined to provide proper insight and also a common language.

The FEI is defined as the activities that precede the structured New Product and Process Development (NPPD).

Being the activities of the FEI very different from those of the NPPD, as FEI tend to be unpredictable and unstructured, and NPPD is a well-defined and structured process, the New Concept Development (NCD) provides a cyclic approach enabling the alignment between both processes. The substantial difference between FEI and NPPD are compared in Table 8.

Table 8 - Comparison of the FEI and the NPPD processes (Koen, et al., 2001)

	Front End of Innovation (FEI)	New Product Process Development (NPPD)
Nature of Work	Experimental, often chaotic. Difficult to plan Eureka moments.	Structured, disciplined and goal-oriented with a project plan.
Commercialization Date	Unpredictable.	Definable.
Funding	Variable. In the beginning phases, many projects may be "bootlegged," while others will need funding to proceed.	Budgeted.
Revenue Expectations	Often uncertain. Sometimes done with a great deal of speculation.	Believable and with increasing certainty, analysis and documentation as the product release date gets closer.
Activity	Both individual and team in areas to minimize risk and optimize potential.	Multi-functional product and/or process development team.

The NDC model consists of three parts, as shown in Figure 14:

1. the definition of the five key elements of the Front End of Innovation (FEI), such as the identification of opportunities, the analysis of opportunities, idea conception and enrichment, the idea selection, and the concept and technological development;
2. the drivers of the five FEI elements, are mostly driven by the leadership and culture of the organization;
3. the factors of influence, such as the capacity of the company, the adopted business strategies, and the external world (distribution channels, customers, and competitors). These influencing factors directly affect the FEI, the NPPD and also commercialization.

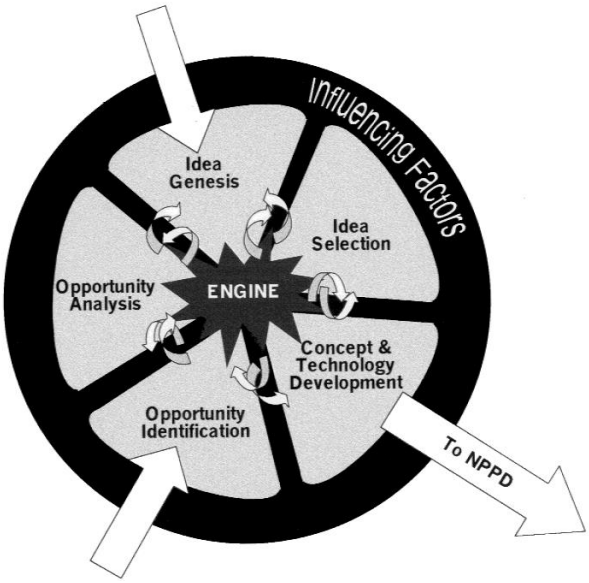


Figure 14 – The New Concept Development Model (Koen, et al., 2001)

The five key elements of New Concept Development can be applied to PdM to elucidate the method to design a market-driven solution:

1. **identification of opportunities:** any manufacturing company has to struggle with equipment maintenance, implicit maintenance costs and equipment downtime while servicing. Reactive maintenance can evolve into a chaotic process, never knowing when equipment breaks down due to lack of maintenance, PvM can be very costly as a consequence of servicing equipment at regular intervals. An alternative solution, between the two formers, could be envisioned as more cost and time effective, in this case, PdM;
2. **the analysis of opportunities:** the present-day manufacturing equipment must be evaluated in terms of the readiness of devices or sensors that could provide real-time information on the status of the working conditions of wear-prone components. Also, the willingness of manufacturing companies to change their maintenance mindset and processes, to trust PdM algorithms to monitor equipment and to be fully capable of

delivering trustworthy predictions. Not identifying unnecessary maintenance, nor misinterpreting urgent maintenance evidence;

3. **idea conception and enrichment:** several methods of delivering PdM were envisioned, analysing sensor raw data and using data mining and artificial intelligence algorithms compared with historical fault and non-fault scenarios, analysing raw sensor data and comparing it with standardized manufacturer tolerances. Using ontologies to model the context of the industrial environment, sensors, time intervals, situations, resources, and conduct reasoning with collected structured data to evaluate potential fault conditions;
4. **the idea selection:** to select one of the ideas, research was performed in **State of the Art**, to adopt one of the ideas that effectively could lead to developing a solution delivering the expected results. The idea of the ontology approach was the selected one;
5. **concept and technological development**: this resulted in this dissertation, by developing an ontology for predictive maintenance that describes the field of application in the industrial manufacturing environment.

In conclusion, the utilization of the NCD's five key elements enabled a structured process:

1. starting with the identification of an opportunity to solve an existing problem;
2. the analysis of the opportunity in terms of its applicability to the manufacturing industry;
3. the idea conception, with the consideration of several approaches that could provide a solution;
4. the selection of an idea based on the research of technologies suited to deliver a solution;
5. finishing with the structured, disciplined and objective-oriented development process represented by the present dissertation.

5.2 Value Proposition Canvas

In any project, to understand the customers' needs and how to satisfy them, the Value Proposition Canvas (VPC), developed by Alex Osterwalder (Osterwalder, 2021), provides a powerful tool to schematize the main ideas (Pijl, 2016). This tool compares what the proposed solution has to offer, and how it contributes to solving the customer's problems.

So, to have an initial view of the solution, a VPC was elaborated, to specify exactly which type of service is to be offered and to assess from the customer's point of view which problem is intended to be solved and which gains the customer can obtain.

Likewise, all the positive outcomes of the service that translate into an immediate gain or satisfaction of the customer were listed, as well as all the components that make life easier for the customer. In this way, it was possible to determine that the service represents an added value for the customer, thus resulting in an attractive and appealing solution.

The VPC permits an overview of the present gains and pains the customer experiences and compares them with the gain creators and pain relievers offered by the solution.

The Value Proposition Canvas for the solution to be developed is presented in Figure 15, based on Alex Osterwalder's template [5].

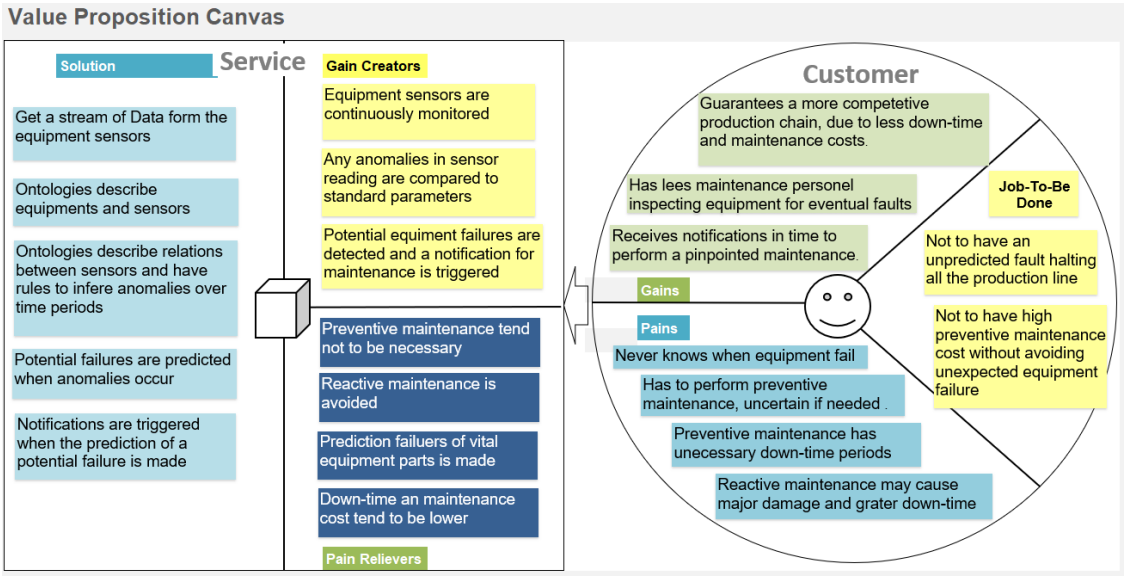


Figure 15 – Value Proposition Canvas

[5] https://neoschronos.com/templates/

The VPC proposes a solution by creating gains or added values for the customers and implements pain relievers to ease the customer's pains or problems, the overall objective is to enable the customer to have his job done.

5.3 Quality Function Deployment

Having in focus the obtainment of the highest possible quality of the solution, according to the previously gathered customer's needs, the Quality Function Deployment (QFD) model was used (Warwick Manufacturing Group, 2007) to perform this analysis. The QFD model is a method that assists in the process of developing a product or service.

In the case of this dissertation, QFD will assist in the development of a service, namely the elaboration of an Ontology-approach of PdM. The first step of this method is to carefully organize the priorities of the product or service, in the following way:

1. **Customer requirements:** What the customer specifically specified as the expected outcomes of the service to be delivered;
2. **Functional Requirements**: What does the solution implement to fulfil the requirements;
3. **Evaluation of the relevance or importance of each requirement to the customer.** A value in the range from [1,9] is applied to each requirement, being 1 the lowest, and 9 the highest;
4. **Obtaining the Relative Importance Weight of each functional requirement.** This value is obtained by the summation of the relation of the importance of customer requirements versus functional requirements. This is a calculated value;
5. The **Direction of Improvement for each functional requirement** should be established to stipulate if each of these requirements is to be maximized, minimized or just to attain the desired target in terms of performance;
6. **Create a Correlation or Relationship Matrix** to highlight the relationships between the customer requirements and the functional requirements.

Once the priorities have been defined, the second step is to build the QFD's House of Quality or Chart of Quality, where all the gathered evaluations, expectations and outcomes are compiled and represented in one single chart.

The guidelines to design the House of Quality are according to Warwick Manufacturing Group (Warwick Manufacturing Group, 2007) and are presented in Figure 16.

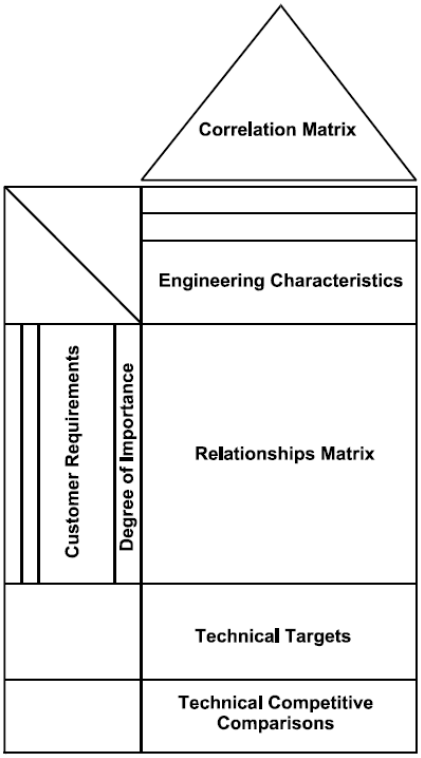


Figure 16 – QFD House of Quality (Warwick Manufacturing Group, 2007)

In Figure 17 the metrics used for correlations, relationships and direction of improvements are detailed.

Correlations		Relationships		Direction of Improvement	
Strong Positive	◉	Strong	●	Maximize	▲
Positive	○	Moderate	○	Target	◇
Negative	✕	Weak	▽	Minimize	▼
Strong Negative	✱				
No Correlation					

Figure 17 – QFD Metrics

The perceived customer's requirements, to provide a solution, are listed below:

1. Data from multiple sensors per equipment must be collected;
2. Multiple sensors per equipment must have established relationships;
3. Multiple sensors per equipment must be reasoned together;
4. Potential failures must be identified in the earliest stages;
5. Potential failures could enable the trigger of a notification;
6. Preventive maintenance of components monitored by sensors should be redundant;
7. False Positives of maintenance should not occur.

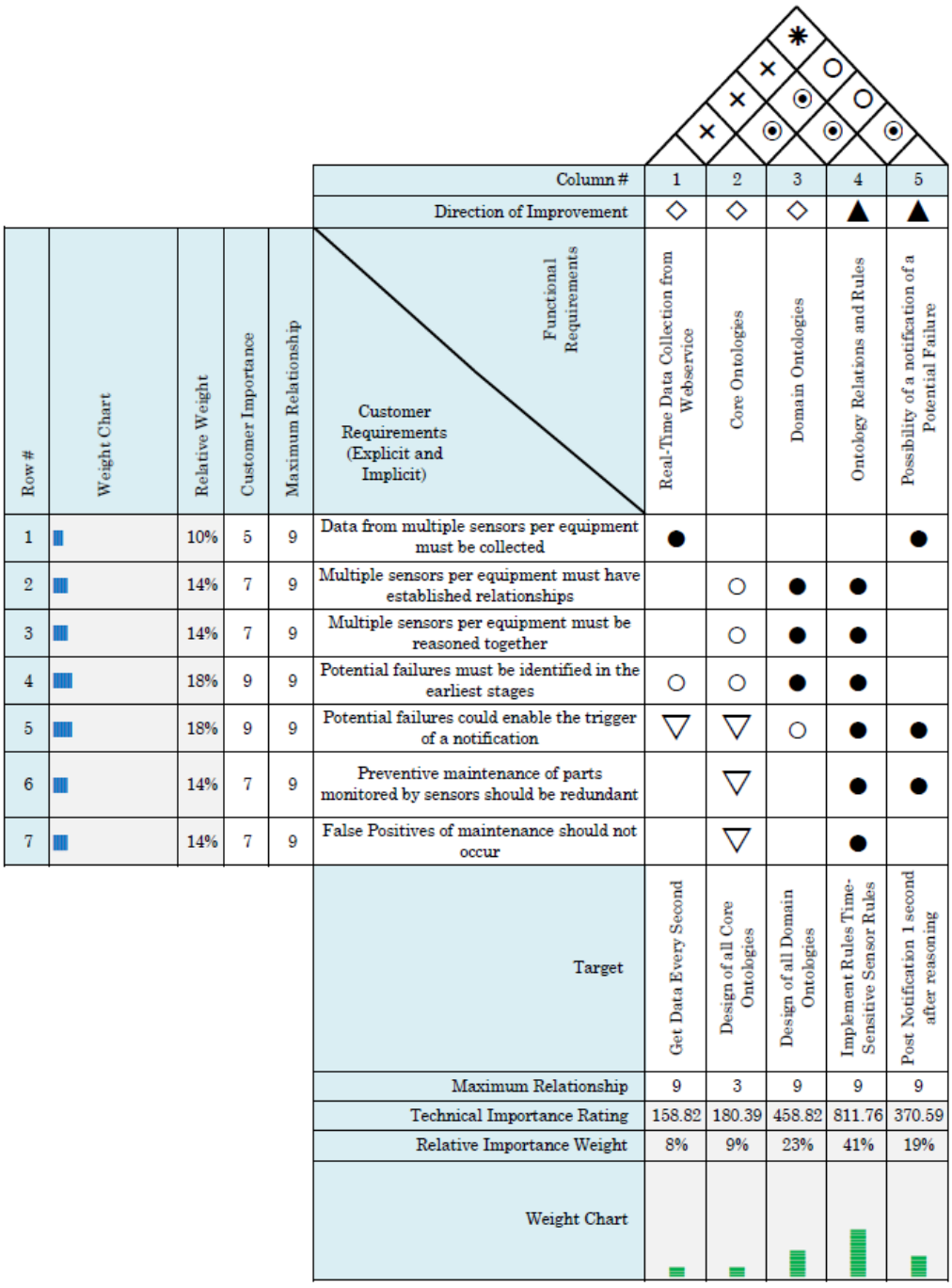
Each of these requirements was assigned a significance value.

The functional requirements to assure the customer’s requirements were identified as stated below:

1. Real-Time Data Collection from a Web Service;
2. Core Ontologies;
3. Domain Ontologies;
4. Ontology Relations and Rules;
5. Possibility of a notification of a Potential Failure.

In this case, for each of these requirements, an importance value was assigned, in terms of relevance to solving the customer’s requirements.

The House of Quality for the solution to be developed is presented in Figure 18, based on Warwick Manufacturing Group's template [7]:



Row #	Weight Chart	Relative Weight	Customer Importance	Maximum Relationship	Customer Requirements (Explicit and Implicit) \ Functional Requirements	1 Real-Time Data Collection from Webservice	2 Core Ontologies	3 Domain Ontologies	4 Ontology Relations and Rules	5 Possibility of a notification of a Potential Failure
					Column #	1	2	3	4	5
					Direction of Improvement	◇	◇	◇	▲	▲
1		10%	5	9	Data from multiple sensors per equipment must be collected	●				●
2		14%	7	9	Multiple sensors per equipment must have established relationships		○	●	●	
3		14%	7	9	Multiple sensors per equipment must be reasoned together		○	●	●	
4		18%	9	9	Potential failures must be identified in the earliest stages	○	○	●	●	
5		18%	9	9	Potential failures could enable the trigger of a notification	▽	▽	○	●	●
6		14%	7	9	Preventive maintenance of parts monitored by sensors should be redundant		▽		●	●
7		14%	7	9	False Positives of maintenance should not occur		▽		●	
					Target	Get Data Every Second	Design of all Core Ontologies	Design of all Domain Ontologies	Implement Rules Time-Sensitive Sensor Rules	Post Notification 1 second after reasoning
					Maximum Relationship	9	3	9	9	9
					Technical Importance Rating	158.82	180.39	458.82	811.76	370.59
					Relative Importance Weight	8%	9%	23%	41%	19%
					Weight Chart					

Figure 18 – QFD: House of Quality for the solution

[7] https://www.coursehero.com/file/43145460/Full-HOQ-091xlsx/

For the customer, the features related to que proper identification of the several equipment sensors, the way they are related to each other, are among the requirements he values most, having by this means a way to describe the equipment working conditions, enabling, the delivery of the outcomes he expects, that potential failures must be identified in the earliest stages.

In terms of functional requirements, the development of Domain Ontologies and the implicit Ontology Relations and Rules have predominant importance to fulfil the expected customer's requirements.

Additionally, the relationships between the customer requirements and functional features were established in the Relationship Matrix. The Maximum Relationship and the Relative Importance Weight were also calculated by multiplying the value of the relevance, for the customer, by the degree of importance relative to the functional requirements.

5.4 TOPSIS decision method

TOPSIS (Technique for Order Preference by Similarity to Ideal Solution) is a numerical method for multi-criteria decision-making. This is a widely applicable method with a simple mathematical model. This method is used when a decision considering multiple alternatives with multiple criteria is needed. (Pavić & Novoselac, 2013).

The TOPSIS method defines that a chosen solution should be as close to the positive ideal solution as possible and as far away from the negative ideal solution as possible (Hwang, Lai, & Liu, 1993).

To have a sustainable decision, if Ontology-Based PdM is by any means a better approach than any other, to perform equipment maintenance, four types of approaches of maintenance are evaluated and compared using the TOPSIS method:

1. Reactive maintenance
    a. Maintenance is only performed after the fault occurs;
    b. Equipment mostly stops working;
    c. Equipment may have suffered several damages;
    d. Very high cost due to the extension of damages;
    e. Very high downtime periods.
2. Preventive maintenance
    a. Maintenance is performed at regular intervals;
    b. Every wear-prone component is substituted, even if still in working conditions;

   c. Very high cost due to the extension of components substituted;
   d. Very high downtime periods.

3. Predictive maintenance using Datamining algorithm over Historic Data
   a. Maintenance is performed considering predictions made over historic data using data mining algorithms;
   b. A huge amount of data must be processed;
   c. Availability of Historic Data is fundamental;
   d. Very low maintenance costs due to early diagnosis;
   e. Very low downtime due to early diagnosis.

4. Predictive maintenance based on Domain Ontologies reasoning
   a. Maintenance is performed considering prediction made using domain ontologies and reasoning;
   b. A scarce amount of data must be processed;
   c. Availability of Historic Data is irrelevant;
   d. Very low maintenance costs due to early diagnosis;
   e. Very low downtime due to early diagnosis.

That evaluation is based on several criteria that are key factors for more efficient maintenance.

1. Maintenance Cost in terms of components and labour;
2. Downtime to perform the maintenance;
3. Processing time needed to evaluate a potential fault;
4. Dependency of Historic Data to perform the prediction.

Observing the selected approaches and criteria some very distinct differences between the mentioned approaches are observable.

Needing to choose the best approach for a possible solution, it is mandatory to have a criteria definition and a specification of weight association to each criterion.

Representation of criteria and approaches is made in Table 9.

Table 9 - Comparison of possible maintenance approaches

	Criteria			
Approaches	Maintenance Costs	Downtime	Processing Time	Historic Data Dependency
Reactive	Very High	Very High	Very High	Irrelevant
Preventive	High	High	Very High	Irrelevant
Predictive Data Mining	Very Low	Very Low	Medium	Very High
Predictive Ontologies-Based	Very Low	Very Low	Very Low	Irrelevant

Considering the challenge to perform a calculated decision, rather than one based just on observation, the evaluation is to be done using the TOPSIS method, which supports making decisions and evaluating choices in multi-criteria decision situations. This method considers three types of criteria:

1. Qualitative benefit attributes/criteria;
2. Quantitative benefit attributes;
3. Cost attributes or criteria.

Firstly, a decision matrix (Matrix ***m*** x ***n***) was elaborated where ***m*** = 4 approaches, ***n*** = 4 criteria. Having $\mathbf{X}_{ij}$ score of options ***i*** with respect to criterion ***j.***

Secondly, options ***i***, criteria ***j*** and weights $\boldsymbol{W_j}$ were enumerated

1. Options ***i*** = {Reactive, Preventive, Predictive, Data Mining, Predictive Ontologies-Based};
2. Criteria ***j*** = {Maintenance Costs, Downtime, Processing Time, Historic Data Dependency};
3. Weights for each criterion $\boldsymbol{W_j}$ = {0,25, 0,35, 0,25, 0,15}.

The weights $\boldsymbol{W_j}$ were considered to minimize maintenance costs and downtime and to minimize the dependency on historic data as well as the maximization of processing time.

Thirdly, an equivalent of the empirical scores to numerical values was performed, represented in Table 10.

Table 10 - Score scale

Irrelevant	Very Low	Low	Medium	High	Very High
6	5	4	3	2	1

And finally, the previous established, options, criteria, weights, and scores enabled the creation of a TOPSIS matrix, represented in Table 11.

Table 11 - TOPSIS Multi-criteria Decision Matrix

Weights	**0,25**	**0,35**	**0,25**	**0,15**
	Criteria			
Options	Maintenance Costs	Downtime	Processing Time	Historic Data Dependency
Reactive	1	1	1	6
Preventive	2	2	1	6
Predictive Data Mining	5	5	3	1
Predictive Ontologies-Based	5	5	5	6

Once the TOPSIS matrix is complete, the calculation process is performed:

- Step 1 (a): calculate (1) for each column;
- Step 1 (b): divide each column by (1) to get $r_{ij;}$
- Step 2: multiply r_{ij} by the weight w_j to get $v_{ij.}$

$$(\Sigma x^2_{ij})^{1/2} \qquad (1)$$

Table 12 – TOPSIS weighted normalized decision matrix

	Criteria			
Options	Maintenance Costs	Downtime	Processing Time	Historic Data Dependency
Reactive	0,0337	0,0472	0,0417	0,0862
Preventive	0,0674	0,0944	0,0417	0,0862
Predictive Data Mining	0,1685	0,2360	0,1250	0,0144
Predictive Ontologies-Based	0,1685	0,2360	0,2083	0,0862

- Step 3 (a): calculate the ideal solution A*;

Table 13 – TOPSIS ideal Solution

	Criteria			
	Maintenance Costs	Downtime	Processing Time	Historic Data Dependency
Ideal solution A* (minimum)	0,1685	0,2360	0,2083	0,0862

- Step 3 (a): calculate negative ideal solution A';

Table 14 – TOPSIS negative ideal solution

	Criteria			
	Maintenance Costs	Downtime	Processing Time	Historic Data Dependency
Negative ideal solution A' (minimum)	0,0337	0,0472	0,0417	0,0144

- Step 4 (a): calculate separation from ideal solution A*, as in (2) and in Table 15;

$$S_i^* = [\ \Sigma\ (v_j^* - v_{ij})^2\ ]^{½} \qquad (2)$$

Table 15 – TOPSIS separation from ideal solution;

Options	S_i^*
Reactive	0,2857
Preventive	0,2409
Predictive Data Mining	0,1100
Predictive Ontologies-Based	0,0000

- Step 4 (b): calculate separation from negative ideal solution A', as in (3) and in Table 16;

$$S_i' = [\ \Sigma\ (v_j' - v_{ij})^2\ ]^{½} \qquad (3)$$

Table 16 – TOPSIS separation from a negative ideal solution

Options	S_i'
Reactive	0,0718
Preventive	0,0923
Predictive Data Mining	0,2465
Predictive Ontologies-Based	0,2945

- Step (5): Calculate the relative closeness to the ideal solution C_i^*, as in (4) and in Table 17.

$$C_i^* = S'_i\ /\ (S_i^* + S'_i) \qquad (4)$$

Table 17 – TOPSIS relative closeness to the ideal solution

Options	C_i^*	
Reactive	0,2009	Worst
Preventive	0,2770	
Predictive Data Mining	0,6914	
Predictive Ontologies-Based	1,0000	Best

The results are shown in Table 17, revealing that the worst solution to be adopted is Reactive Maintenance, where a complete absence of scheduled or planned maintenance is performed, and on the opposite end, the best solution is an Ontology-Based PdM. The main factors contributing to this conclusion were the non-dependency of historic data to evaluate a potential failure and thus the more efficient processing time to evaluate a potential fault.

The step-by-step TOPSIS decision process, systematically detailed and represented in Table 9 to Table 17, demonstrated and concluded, that considering the given criteria and weights for each criterion the ideal solution to be chosen, is the Ontology-Based PdM, among those evaluated.

6 Solution Design

In this chapter, a practical application of the ontology to be developed in this dissertation is presented and explained in detail.

The purpose is to develop an ontology for predictive maintenance that describes the scope and field of application by describing real-time sensor data; a practical use of this ontology would be in an application that classifies and transforms data from sensors, generating new knowledge in the form of triples so that they may serve as input to ML and DM algorithms.

This solution is documented with the requirements and constraints, by a Use-Case Diagram, a Component Diagram, and the System Sequence Diagrams (SSD) of the use cases.

6.1 Requirements and Constraints

In this section, the requirements and specifications needed to use ontologies applied in the field of PdM are identified, as well as the constraints that may endanger a successful solution implementation.

6.1.1 Requirements

Requirements will be detailed in this section according to three major topics:

1. Data Collection - the means to collect data from the equipment;
2. Data Format - how to perform the interpretation of the collected data;
3. Domain Ontology - the design of the domain ontology.

6.1.1.1 Technologies and Tools

In terms of technologies only two requirements were identified, the use of Ontologies, described in section 2.1.2 Ontology Languages and the Java Programming Language described in section 3.2.1 Java SE and Apache NetBeans.

6.1.1.2 Data Collection

To be able to evaluate data from sensors monitoring vital components of manufacturing equipment, independently of the evaluation to be performed, collecting the data in real-time must be achievable. PdM must have the ability to evaluate what is happening at a precise moment. Analysing collected data with delays means that the predictions may relate to events that occurred in the past and thus, the prediction is not accurate regarding the potential fault to occur.

In terms of requirements, some assumptions must be granted, namely, that data is collected with no delays or minimum delays, and that the data is already presented in a readily usable format, not needing transformations or uniformization, consuming precious processing time before its usage. The requirements identified regarding data collection are as follows:

1. the acquisition of the data is to be performed by enquiring a Web Service using Hypertext Transfer Protocol (HTTP) GET method;
2. the HTTP GET Method must specify by the usage of query parameters the equipment targeted to be surveyed in terms of PdM;
3. the HTTP GET Method returns a JavaScript Object Notation (JSON) response with a collection of sensor data, identifying: each sensor, the list of values obtained, and the time of capture (e.g. with a timestamp).

Sample data is obtained from the PIANiSM (PIANiSM, 2020) project through a Web Service, as shown in Figure 19.

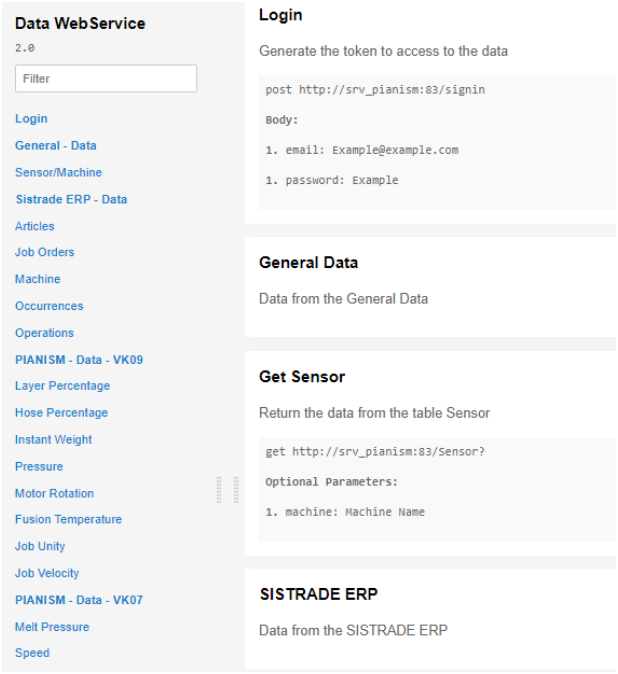


Figure 19 - PIANiSM Web Service

This Web Service provides a set of endpoints to obtain data from the Sistrade ERP, containing:

1. Data from the ERP relative to manufacturing processes:
    a. manufacturing orders;
    b. manufacturing operations;
    c. maintenance occurrences.
2. Data collected from the equipment's sensors and made available by the ERP:
    a. equipment involved in the manufacturing process;
    b. equipment's configurations, like heads or layers;
    c. maintenance occurrences;
    d. sensors data providing a real-time reading of feed rate, fusion temperature, instant weight, job velocity, melt pressure, motor current, motor rotation, pressure, speed, and other sensors.

Each set of data is provided by a dedicated endpoint and thus does not have any correlation, like those exemplified in the following list:

    a. get http://srv_pianism:83/JobVelocity?
    b. get http://srv_pianism:83/MeltPressure?

c. get http://srv_pianism:83/FusionTemperature?

6.1.1.3 Data Format

Collected data must be properly formatted to be interpreted. The data to be analysed and reasoned with must be submitted to a previous homogenization process, assuring that independently of the equipment type, the sensor type or the type of measure the sensor performs, the data is adequately normalised.

The requirements identified regarding data interpretation are as stated below:

1. data obtained in JSON format must be properly parsed in pairs of *Key/Value* by sensor type, the ***Key*** represents the timestamp of the sensor and the ***Value*** the readings itself;
2. data obtained has also to be catalogued and stored to create an inventory of available data to serve as a source of the application of ML and DM algorithms.

The data is obtained in the body of the Http:get request in JSON notation, like the sample shown in Figure 20 andFigure 21.

```
[
    {
        "timestamp": "2021-09-10T00:00:03.110Z",
        "rawData": 35.8983154296875
    },
    {
        "timestamp": "2021-09-10T00:00:08.110Z",
        "rawData": 36.04350662231445
    },
    {
        "timestamp": "2021-09-10T00:00:13.110Z",
        "rawData": 35.952762603759766
    },
    {
        "timestamp": "2021-09-10T00:00:18.110Z",
        "rawData": 36.152400970458984
    },
```


Figure 20 - Job Velocity Data

```
[
    {
        "MachineCode": "VK07",
        "MachineDescription": "Extruder 7 Layers",
        "Section": "S01",
        "Extrusion_heads": "A,B,C,D,E,F,G"
    },
    {
        "MachineCode": "VK09",
        "MachineDescription": "Extruder 9 Layers",
        "Section": "S01",
        "Extrusion_heads": "A,B,C,D,E,F,G,H,I"
    },
    {
        "MachineCode": "VI04",
        "MachineDescription": "Printer",
        "Section": "S02",
        "Extrusion_heads": "n/a"
    }
]
```


Figure 21 - Equipment Data

6.1.1.4 Domain Ontology

Monitoring industrial equipment to perform PdM requires a description of what is monitored:

1. the sensor data;
2. what does the monitoring action measure;
3. additional information provided by the ERP, e.g. processes in execution, the process stage, the material being processed, etc.;
4. how the data gathered from sensors is considered in the range of accepted tolerances;
5. the relation of the sensor data and the process stage in execution.

Depending on the equipment type, a sensor measures a feature or variable related to a component that must be evaluated independently of the readings of other sensors, but cases can occur where two related sensors must be evaluated in aggregation. The domain ontology to be developed must be able to describe the equipment's sensors, the expected readings and the corresponding periods, as well as their relationships.

The requirements identified regarding the development of a domain ontology for PdM are as stated below:

1. to represent a sensors as a class;
2. to define the attributes, properties, and features for the sensor class;
3. to represent the logical relations between the attributes of the sensor class;
4. define date/time intervals restrictions;
5. infer rules over the sensor class and date/time restrictions.

6.1.2 Constraints

The main constraint enclosed in this work is the dependency on data supplied by equipment in real working conditions, such as an industrial or manufacturing environment. This means that the access to real-life data collected from equipment sensors and data from the manufacturing processes is of paramount importance. Assuming that the previous scenario is secured, the collected data must be processed to eliminate any invalid readings and cleaned of all unnecessary values forwarded to the companies' ERP. The cleaned sensor data must be bundled by equipment and only then transformed into a ready readable JSON structure and made available through an HTTPS Web Service.

Of great importance is the availability of data sets describing equipment working in healthy and stable conditions and describing equipment working in malfunctioning or unstable conditions. To exemplify, there is no empirical way to provide correct working parameters for:

1. a ball bearing's optimal or harmful working vibration, nor the exact time slices a ball bearing can support a peak vibration;
2. an extrusion head's optimal or harmful working temperature, nor the exact time slices an extrusion head can support a peak temperature.

To enforce the identified constraints, the absence of real-time data or equipment specifications would ultimately lead to testing the work with mock data, and establish that a functional prototype was developed, but could fail into proving the correct behaviour in real-life environment.

6.2 Use Cases

In this section, the use-case diagram identifying the use cases of a component, named **PdM Ontology Component**, which would account for a practical application of the Domain Ontology is presented in Figure 22. In the presentation of these use cases, Time-Sensitive Sensor Data and Manufacturing Process Data, is referenced shortly as Time-Sensitive Data.

The identified use cases are:

1. **UC1** – Get Time-Sensitive Data from the ERP, in a continuous loop;
2. **UC2** – Performs the transformation of the Time-Sensitive Data using the Domain Ontology's description and triggers ML and DM algorithms to detect anomalies;
3. **UC3** – Sends Potential Failure Notification to the ERP, alerting that a potential failure was identified.

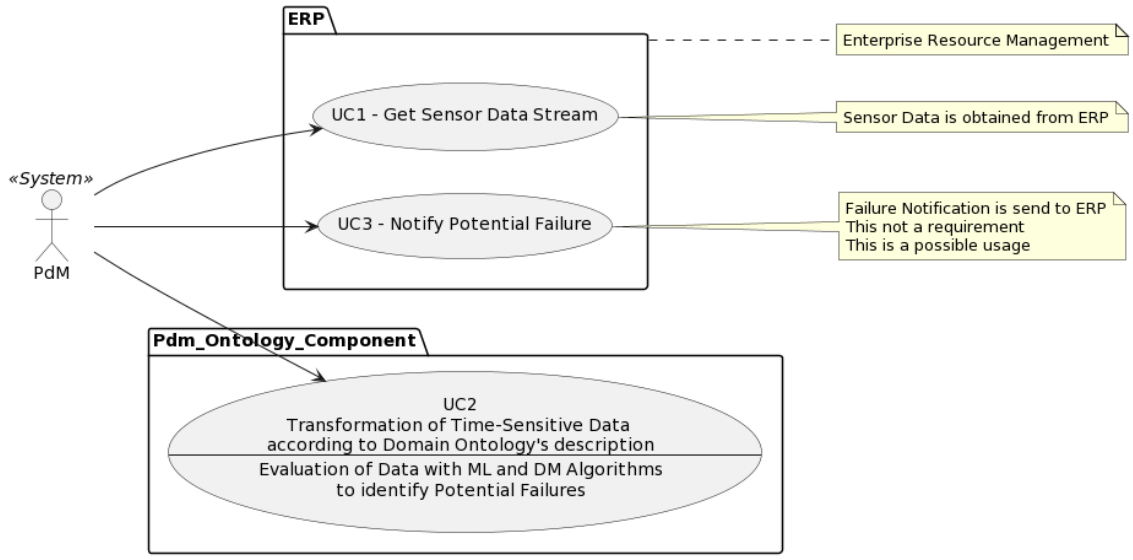


Figure 22 – Use-Case Diagram

6.3 Component Diagram

The component diagram shown in Figure 23 illustrates how the **PdM Ontology Component** interacts with the customer's system.

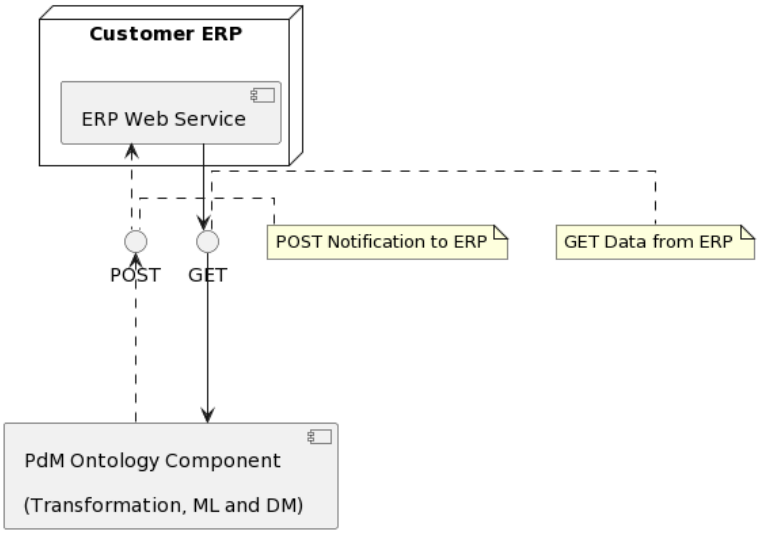


Figure 23 – Component Diagram

The **PdM Ontology Component** interfaces with the customer's ERP component, by performing cyclic HTTP GET requests to obtain the Time-Sensitive Data.

Time-Sensitive Data is transformed according to the Domain Ontology description and evaluated through ML and DM to detect anomalies leading to potential failures.

In case a potential failure is identified the customer's ERP component is notified by an HTTP POST, detailing the potential failure identified.

6.4 System Sequence Diagrams

6.4.1 UC1 – Get Time-Sensitive Sensor Data from the ERP

Brief Format: The **PdM Ontology Component** requests Time-Sensitive Data from the ERP. The ERP returns the requested Data. The data is provided to the transformation, ML and DM algorithms.

UC1's System Sequence Diagram is presented in Figure 24.

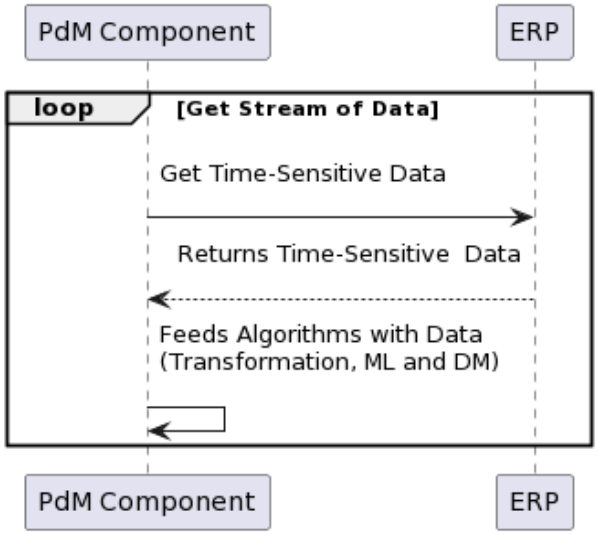


Figure 24 – UC1 - System Sequence Diagram

Full Format

Main Actor: PdM Ontology Component

Stakeholders and their interests: The PdM Ontology Component needs the Time-Sensitive Data to perform the PdM evaluation

Pre-conditions: The ERP must have an endpoint where Time-Sensitive Data can be requested.

Post-Conditions: Requested Time-Sensitive Data is received.

Main success scenario (or basic flow):

1. The PdM Ontology Component requests Time-Sensitive Data from the ERP, by an HTTP GET;
2. The ERP returns the requested data in the response body

3. The Time-Sensitive Data is forwarded to the transformation and evaluation algorithms;
4. This process loops infinitely to step 1.

6.4.2 UC2 – Performs the Transformation and Evaluation

Brief Format: The **PdM Ontology Component** feeds the transformation and evaluation algorithms with Time-Sensitive Data. Evaluation of received data is performed. Received data can show evidence of potential failures. The algorithm continues the loop of receiving data to reason over.

UC2's System Sequence Diagram is presented in Figure 25.

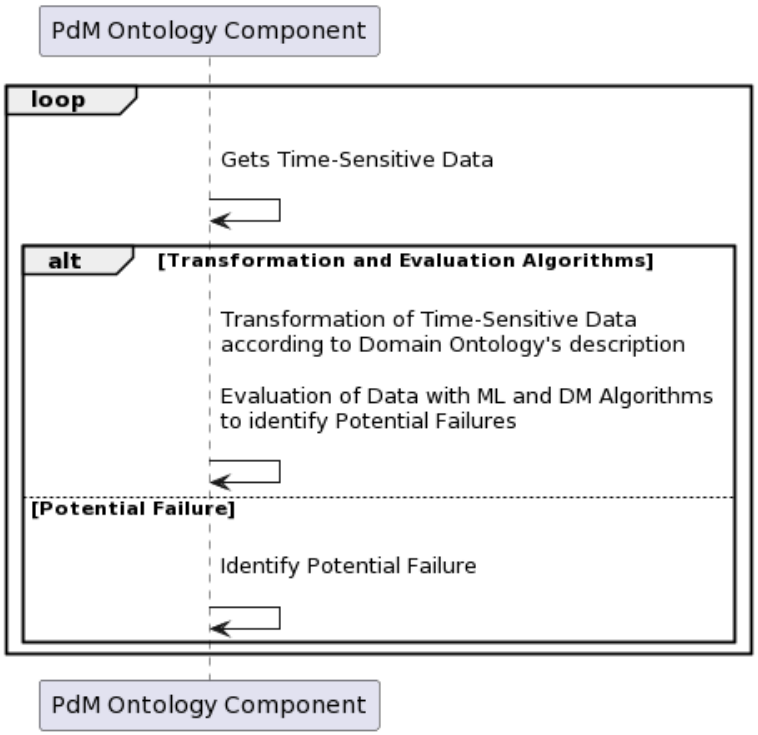


Figure 25 – UC2 - System Sequence Diagram

Full Format

Main Actor: PdM Ontology Component

Stakeholders and their interests: The PdM Ontology Component uses the Time-Sensitive Data to perform the PdM evaluation and to identify if a potential failure may be imminent.

Pre-conditions: The PdM Ontology Component must have received the Time-Sensitive Data.

Post-Conditions: The transformation and evaluation are performed.

Main success scenario (or basic flow):

1. The **PdM Ontology Component** feeds the transformation and evaluation algorithms with Time-Sensitive Data;
2. Transformation and evaluation of Time-Sensitive Data are performed. Received data can show evidence of potential failures;
3. The algorithms continues the loop of receiving data to transform and evaluate.

6.4.3 UC3 – Sends Potential Failure Notification to the ERP

Brief Format: The **PdM Ontology Component** informs of an identified potential failure. A notification is sent to the ERP alerting of a potential failure.

UC3's System Sequence Diagram is presented in Figure 26.

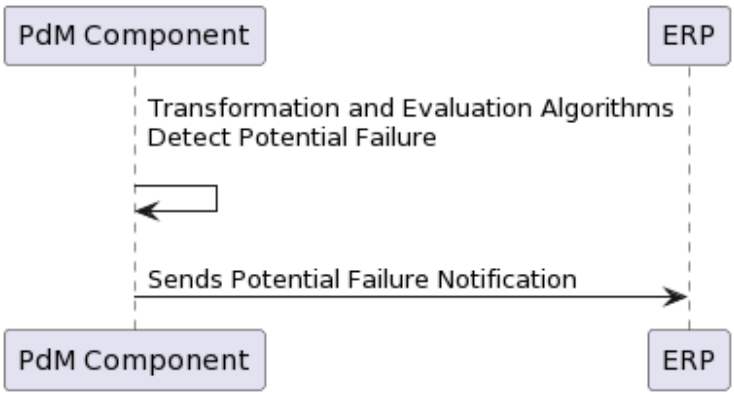


Figure 26 – UC3 - System Sequence Diagram

Full Format

Main Actor: PdM Ontology Component

Stakeholders and their interests: The ERP is notified if a potential failure was identified.

Pre-conditions: The transformation and evaluation algorithms must inform of the identification of a potential failure.

Post-Conditions: The ERP must be notified of the potential failure.

Main success scenario (or basic flow):

1. The PdM Ontology Component informs of an identified potential failure;
2. A notification is sent to the ERP alerting of a potential failure, by an HTTP POST, sending the details in the payload.

7 Ontology Design

In this chapter, a detailed design of the proposed ontology is presented. Firstly, the ontology requirements are specified. Secondly, a review of the already existing and re-usable ontologies in the field of manufacturing processes is made. Thirdly, the design of an ontology required to represent data from PIANiSM (PIANiSM, 2020) is described. Finally, the proposal of a domain ontology bridging the gap between all the former ontologies is made.

7.1 Domain Ontology Requirements

Before the design process of the ontology, it is important to elucidate the reason the ontology is needed, identify the target users, the purpose of the usage and specify the requirements the ontology must accomplish. The process of defining the requirements is covered in the Ontology Requirements Specification Document (ORSD) (Suárez-Figueroa, Gómez-Pérez, & Villazón-Terrazas, 2009), which works as a guide to define the requirements that should be considered by the proposed ontology. The ORSD is defined by enumerating the ontological needs to cover the focused domain: these specifications aren't static, as they are established at the beginning of the development process and continue to be refined as the design process progresses and more detailed needs are identified. The ORDS for the proposed ontology is presented in Table 18.

Table 18 - Ontology Requirements Specification Document

Extrusion Process Domain Ontology Requirements Specification	
1	Purpose
	The need to develop an integrated ontology for predictive maintenance that describes the field of application of the plastic extrusion process and bridges gaps between existing ontologies.
2	Scope
	The ontology must consider all concepts and events involved in the plastic extrusion process, focusing on the description of plastic extrusion equipment and the data collected from the equipment sensors. It should also allow for the description of the data collected from the ERP managing the extrusion process, such as manufacturing orders, manufacturing jobs and occurrences or incidents. Also, the ontology should introduce for each event or data captured in an instant or time lapse the necessary temporal representation of these occurrences by introducing temporal events with properties referencing the beginning and end of the respective time periods.
3	Implementation Language
	Implementation in OWL Language
4	Intended End-Users
	User1: Sistrade, by applying the developed ontology to the PIANiSM project **User2:** Any other extrusion process control system needing interoperability among manufacturing systems, collecting data and knowledge extraction
5	Intended Uses
	Use1. Define a domain ontology that describes the field of application of plastic extrusion manufacturing process and bridges gaps between existing ontologies **Use2.**The developed ontology should be suitable to be utilised for predictive maintenance
6	Ontology Requirements
	a) Non-Functional Requirements
	NFR1. All the Classes, Object and Data Properties, Labels and Comments are to be presented in English **NFR2.** All added Classes, Objects and Data Properties must use appropriate prefixes to distinguish themselves from the ones imported into the existing ontologies
	b) Functional Requirements
	CQ1. Which are the components of plastic extrusion equipment? Heads, Sections, Sensors, etc. **CQ2.** Which kind of sensors are present in each extrusion head? Heat, Voltage, Pressure, Rotation, etc.; **CQ3**. Which Manufacturing Orders are processed by which equipment and when? **CQ4.** Which Manufacturing Operations are performed with which equipment and when? **CQ5**. Which Manufacturing Occurrences happened in which equipment and when? **CQ6.** Which kind of reading occurred at each sensor? **CQ7.** Which was the value read at each sensor and during which time interval the reading was present?
7	Pre-Glossary of Terms

	Extruder ExtrusionHead Sensor Layer	ManufacturingOperation ManufacturingOrder Occurrence	Event Interval Time Instant Time Duration

7.2 Usage of existing Ontologies

The purpose of this work is to develop an ontology for predictive maintenance that describes the field of application and bridges gaps between existing ontologies. Therefore, before the development of the proposed ontology, research was undertaken to look for existing public available ontologies containing concepts that could be reused in the scope of this work. These ontologies were previously described in the State of the Art section of the present document. The chosen ontologies, following research, focused on domains related to manufacturing processes, extrusion machines, industrial equipment sensors, representation of time and time-related concepts, and ontologies with concepts representing data mining processes and entities, namely:

- CDM-Core Ontology
- ExtruOnt Ontology
- SSN Ontology
- Time Ontology
- Onto-DM Ontology
- ASIIO Ontology

7.3 Development of additional Ontologies

Besides the usage of already existing public ontologies, in order to represent concepts corresponding to real-time data gathered from PIANiSM available through a Web Service, an additional ontology was developed to represent the data from PIANiSM. This ontology was named ***OntoPianismErp.***

All the data gathered through the Web Service was carefully analysed and classified to specify the classes, objects and properties to accurately represent it. The result is an ontology that fully represents the scope of the working environment of PIANiSM. With ***OntoPianismErp*** ontology representing the domain of the PIANiSM extrusion process, the data collected from the Web Service had to be stored according to the definition of the developed ontology. As such, a repository using OWL syntax was developed to be used as storage for the Individuals representing the actual collected data. This repository was named ***OntoPianismIndividuals***.

7.3.1 OntoPianismErp Ontology

Name: OntoPianismErp Ontology

Purpose: Ontology providing information about the scope of the PIANiSM (PIANiSM, 2020) project. It describes manufacturing orders, operations and occurrences of the manufacturing process, as well as extrusion machine configuration and sensor readings. The information represents the plastic extrusion manufacturing process.

namespace: http://www.example.org/Pianism/OntoPianismERP.owl

imports: OntoPianismIndividuals

Domain of concepts represented by this ontology:

1. Concepts representing ERP data:
    a. manufacturing orders;
    b. manufacturing operations;
    c. maintenance occurrences.
2. Concepts representing machine and sensor data:
    a. extrusion machines in the manufacturing process;
    b. extrusion machine's configurations, like heads or sections;
    c. temporal events during time periods occurring in extrusion machines and sensors.

7.3.2 OntoPianismIndividuals Repository

This repository is empty at its creation and by itself does not introduce new concepts nor object and or properties – it serves as a container to hold Individuals imported or converted from the WebService, according to the domain ontology to be developed.

Name: OntoPianismIndividuals

Purpose: Repository providing information about PIANiSM (PIANiSM, 2020) Individuals collected from the extrusion manufacturing process. The individuals stored in this repository can be subjected to the application of ML and DM algorithms. Being created using OWL syntax, this repository can be imported by the ***OntoPianismErp*** ontology.

namespace: http://www.example.org/Pianism/OntoPianismIndividuals.owl

7.4 Domain Ontology Structure

The proposed domain ontology, named ***OntoProcessMapping***, bridging the gap between existing public ontologies, is composed of the following ontologies by direct import:

1. **Time**, usage of concepts of the nature of temporal properties, instants, and intervals;
2. **ASIIO**, usage of concepts of the nature of alerts, alarms, types of severities and types of events;
3. **CDM-Core**, usage of concepts of the nature of faults, fault states, symptoms, and components;
4. **SSN**, usage of concepts of the nature of sensor types and properties;
5. **Onto-DM**, usage of concepts of the nature of data mining algorithm execution and implementation of predictive modelling;
6. **ExtruOnt**, usage of concepts of the nature of the description of a plastic extrusion machine, such as extrusion head features, sensor features, and others;
7. **OntoPianismERP**, the auxiliary ontology describing data collected from the PIANiSM ERP (PIANiSM, 2020);
8. **OntoPianismIndividuals**, the repository to store semantic data in the form of individuals.

The domain ontology structure is presented in Figure 27.

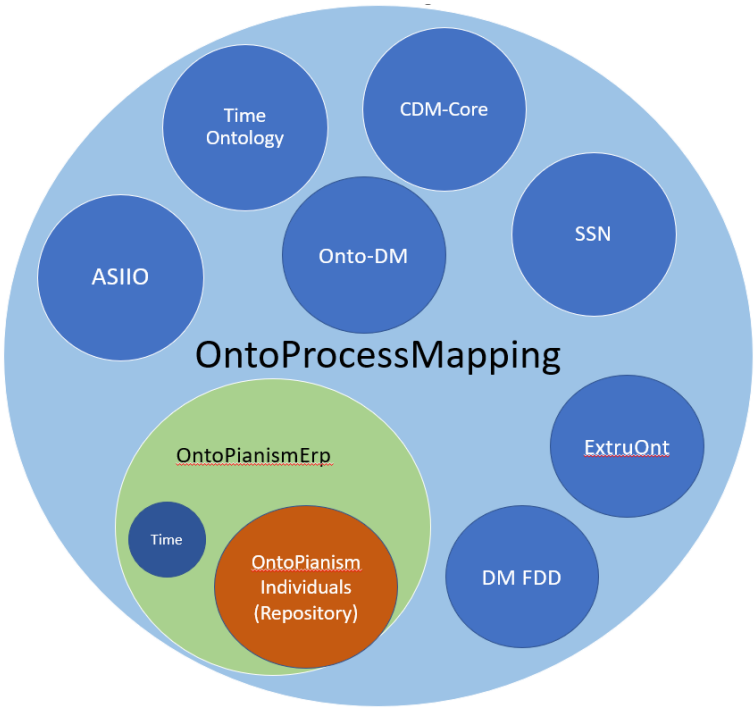


Figure 27 – OntoProcessMapping Ontology Structure

7.4.1 Domain Ontology Design

Previous research demonstrated that no single ontology was yet available to characterize the field of manufacturing processes and the representation of its time-sensitive data. ***OntoProcessMapping,*** was developed to fill this gap using the previously described public available ontologies.

To provide time-related attributes, the **Time Ontology** (OGC & W3C, 2020) was integrated into the domain ontology. Time was developed by W3C, containing definitions of Time Instant to represent the beginning and end of concrete time moments, the Time Interval representing the interval between two instants and Time Duration to represent a numerical value of temporal duration. This ontology defines all the necessary time concepts for this work and is represented in Figure 28.

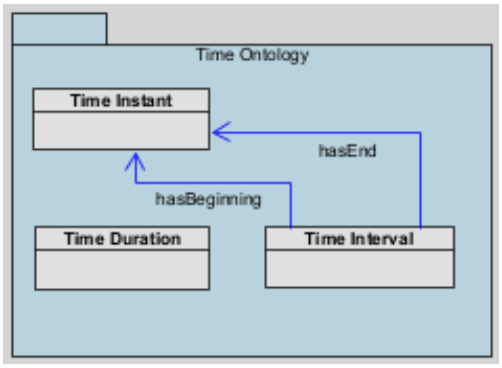


Figure 28 – Concepts of Time Ontology (subsection)

The **ASIIO Ontology** (Aleid & Canito, 2020), used in Airport Security Interoperability, provided a set of concepts also applicable to manufacturing systems – particularly those concerned with Events related to abnormal equipment working conditions. These Events trigger Alerts of different types (Alarm, Warning, Advisory or Info). Furthermore, an Alarm can have different severities (Extreme, High, Medium, and Low). Depending on the Event, several degrees of Criticality conditions can occur, from Normal, Emergency, up to Escalation. Once an Alert is triggered, its Status can be monitored as either Enabled or Disabled. The same structure or flow applicable to Airport Security is observed when a fault occurs in a manufacturing process and consequently triggers an event. The whole subsection of ASIIO covering this field is used as is and included in the domain ontology. A subsection of the ASIIO Ontology covering the previous focussed concepts is presented in Figure 29.

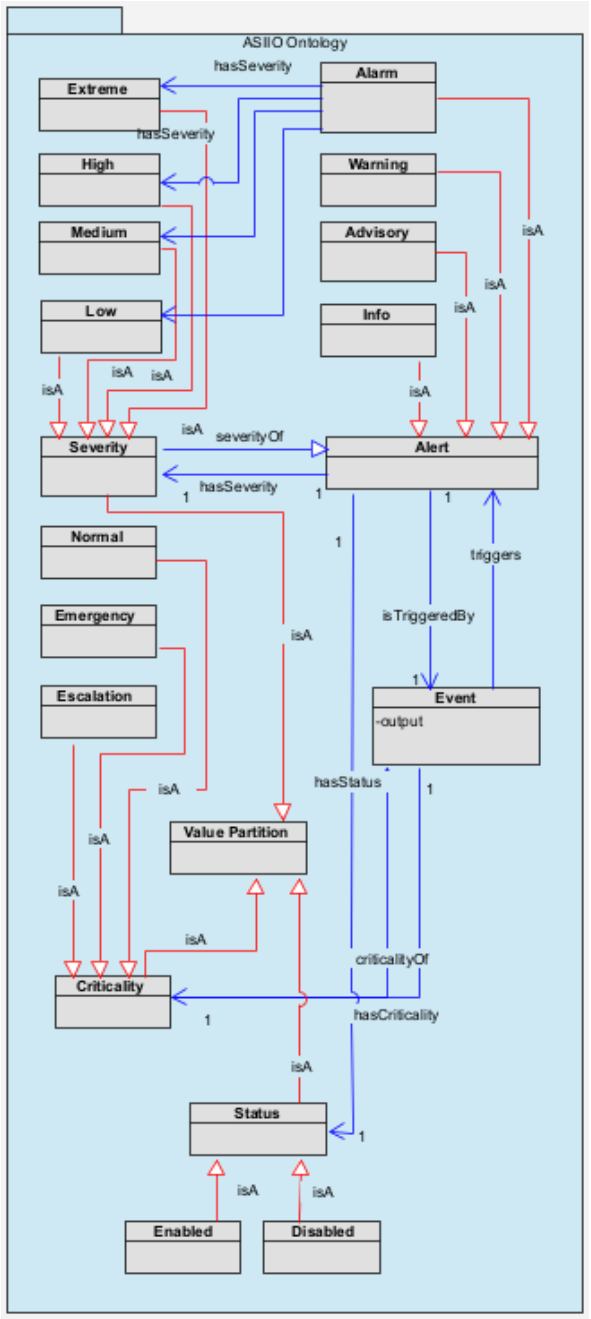


Figure 29 - Concepts of ASIIO Ontology (subsection)

The **CDM-Core Ontology** is the largest, in terms of concepts, publicly available ontology for the manufacturing domain in OWL2 format (Mazzola, Kapahnke, Vujic, & Klusch, 2016). The purpose of this ontology was to balance general manufacturing process applicability and to cover use-case specificity domains, namely automotive exhaust production and metallic press maintenance.

It was successfully used to annotate different aspects of the manufacturing process cycle, such as process models, services, and data streams, and demonstrated to cover all the elicited requirements. The concepts present in this ontology, and re-used in the proposed domain ontology, are used to identify System Conditions and Faults, as well as Component Conditions and Faults. A fault is characterized by a Fault State and a Symptom. The CDM-Core Ontology imports the SSN Ontology also developed by W3C (W3C, 2011), re-using the definition of sensors and observations. Hereby both ontologies are interconnected by Sensors observing specific properties which are manifested in the equipment's components. Thus, sensors are actively monitoring the component's properties (e.g. temperature, pressures, current, flow, etc.), enabling the identification of symptoms leading to probable faults. The CDM-Core and SSN Ontology contribute substantially to the proposed domain ontology, allowing the specification of the role of sensors monitoring components. Both of the ontologies' concepts that were used in this work are presented in Figure 30.

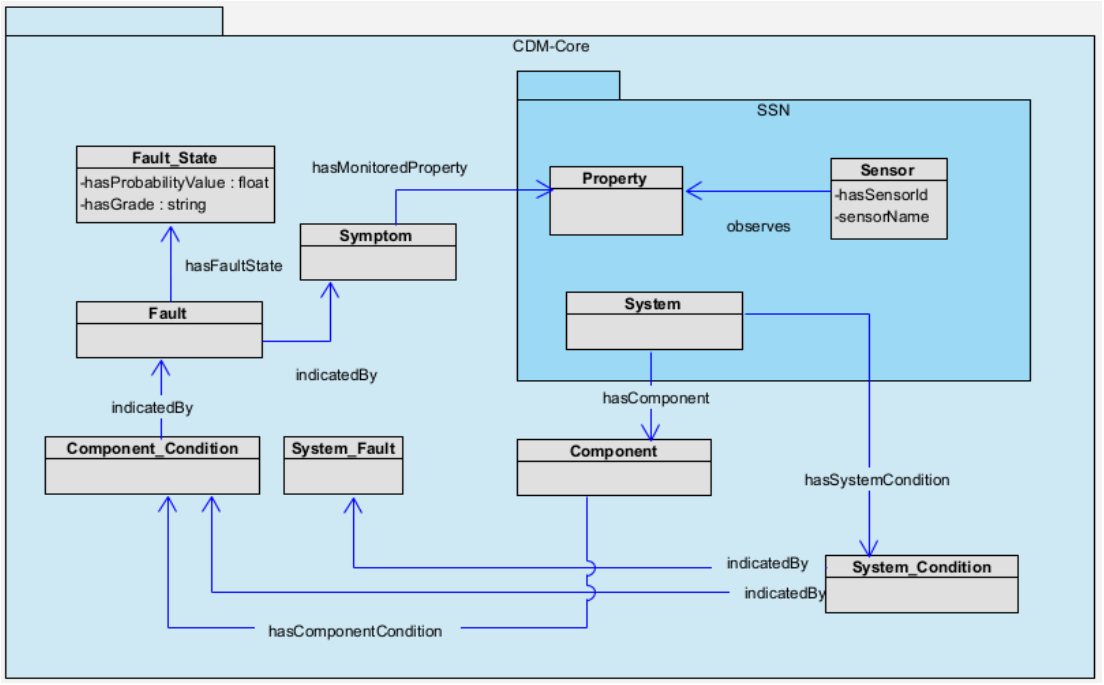


Figure 30 - Concepts of CDM-Core and SSN Ontology (subsection)

The **ExtruOnt Ontology** provides a wide range of terms and concepts related to an extruder, containing classes and properties for explicitly describing extruder components, spatial connections and features, as well as the distinct types of sensors observing the properties of each component, thus enabling the capture of working conditions of each component (Ramírez-Durán, Berges, & Illarramendi, 2019). This ontology is a freely available resource to describe extruder machines and is therefore incorporated into the proposed domain ontology. The ExtruOnt concepts directly used in the domain ontology are presented in Figure 31.

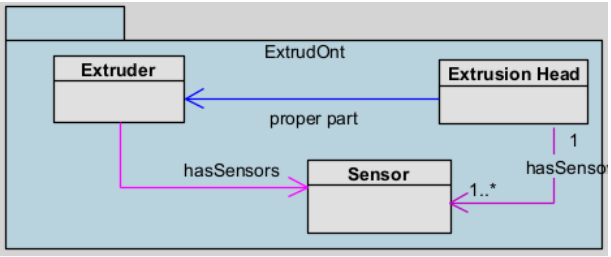


Figure 31 - Concepts of ExtruOnt Ontology (subsection)

The **OntoDM** was developed to represent the different types of structured data uniformly as a general-purpose modular domain ontology for Data Mining (DM) (Panov & Stefan, 2020). OntoDM is composed of three module ontologies: OntoDT, an ontology for datatypes based on ISO standards; OntoDM-core, an ontology for core DM entities to describe mining of structured data; and OntoDM-KDD, an ontology for representing data mining investigations. Each of the mentioned ontology modules can be used independently. The OntoDM ontology was chosen to be incorporated in the proposed domain ontology as it specifies the necessary data mining entities (dataset, generalization, DM algorithm), their specifications (data specification, data mining task) and the DM processes these entities participate in (such as data mining algorithm execution). It also provides the description for data mining structured data, taxonomies of datasets, data mining tasks, and data mining algorithms. The **OntoDM** ontology concepts used in the domain ontology are presented in Figure 32.

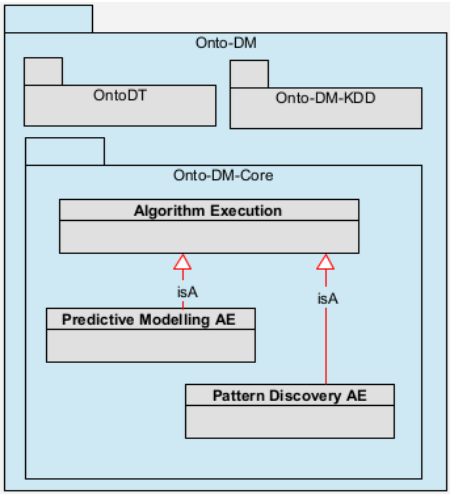


Figure 32 - Concepts of OntoDM Ontology (subsection)

In addition to the previous mentioned publicly available ontologies, a new ontology, **OntoPianismERP**, representing the concepts of the PIANiSM project describing the plastic extrusion process was developed. This ontology describes the concepts that are unique to the PIANiSM project and are not found in any of the previously included ontologies. This ontology includes concepts to describe the extrusion machine as a resource, the manufacturing process triggered by a manufacturing order, and the raw materials to be used in several manufacturing operations. During the manufacturing process, occurrences can be caused by maintenance tasks setting the extrusion machine into a specific status. The detailed implementation of this ontology is described in **8.1 OntoPianismErp - Pianism ERP Ontology Implementation.**

7.4.2 Domain Ontology Diagram

The ***OntoProcessMapping*** ontology represents the complete domain of the developed ontology, including the publicly available ontologies and the **OntoPianismERP** ontology developed specifically to represent the PIANiSM domain of concepts, the complete diagram is shown in Figure 33. To better understand the structure of the ontology, different colours are used to provide a better visual perception:

- The OntoProcessMapping domain ontology is represented in grey;
- The imported public available ontologies are represented in blue;
- The OntoPianismErp ontology developed to provide information about PIANiSM (PIANiSM, 2020) manufacturing processes, represented in yellow.

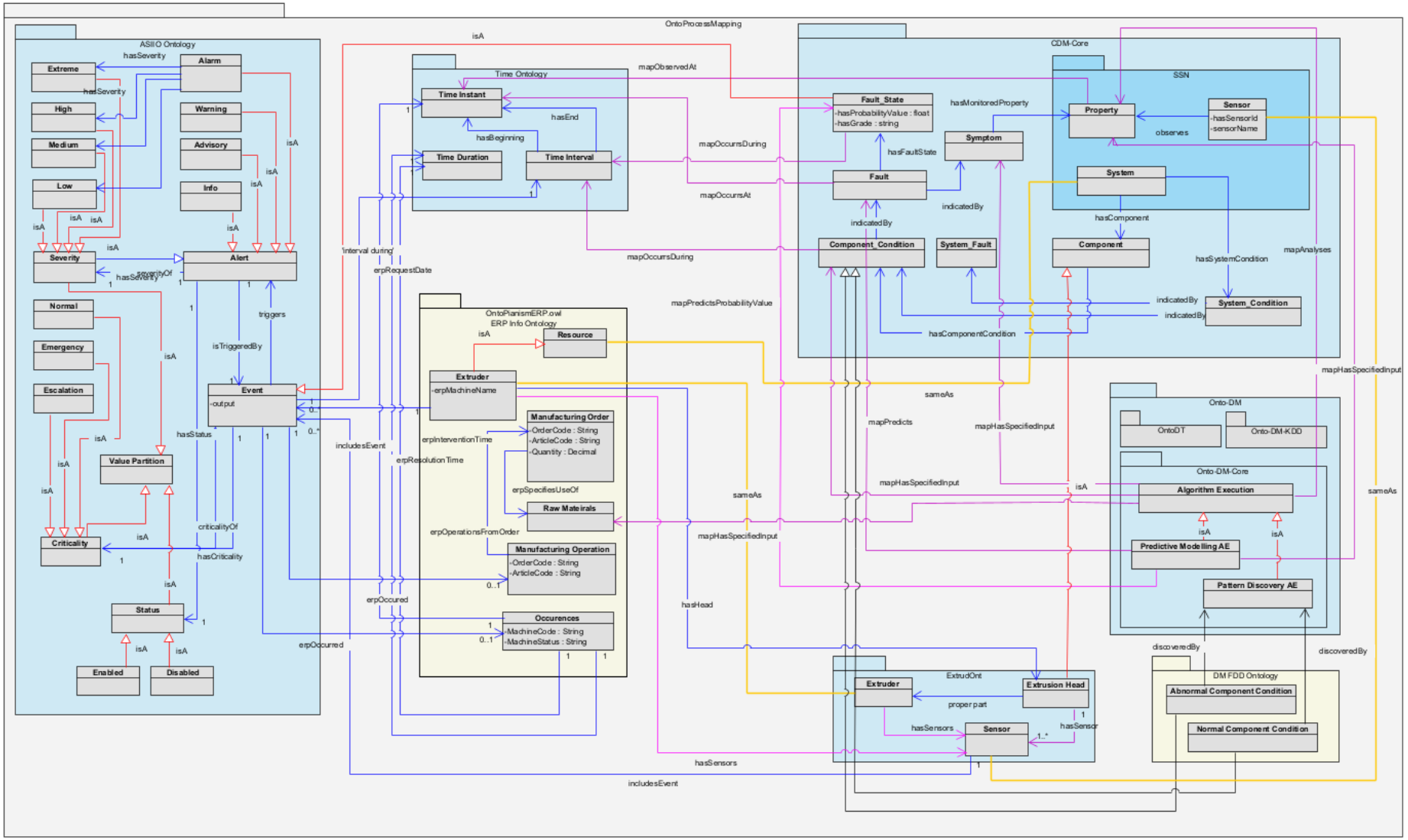


Figure 33 - OntoProcessMapping Ontolo

8 Ontology Implementation

This chapter will describe the implementation of the ***OntoPianismERP*** and the ***OntoProcessMapping*** ontology. For brevity and visual clearness, screenshots of the definitions in Protégé are used. It will include the description of classes representing concepts, the object properties representing relations, and data properties representing concepts' attributes.

8.1 OntoPianismErp - Pianism ERP Ontology Implementation

8.1.1 Classes

The PIANiSM manufacturing process domain was structured and implemented as follows:

The ***ErpPianismThing*** class is the entry point of the ***OntoPianismErp*** implementation and a direct subclass of the ***'OWL:Thing'*** class. This class is the root of all classes in this ontology, grouping the associated concepts and isolating them from other ontologies, either directly imported or present in the ***OntoProcessMapping*** domain ontology. The ***ErpPianismThing*** class' first level hierarchy is presented in Figure 34.

Figure 34 - ErpPianismThing Class

The ***ErpPianismThing*** subclasses were implemented as follows:

The **Manufacturing Operation** is an action performed at a specific time instant on the manufacturing resource, issued from the ERP. The operation specifies the raw materials, the

amount of final product harnessed, the rejected quantities and the interval during operations occurred. The ***ManufacturingOperation*** class is represented in Figure 35.

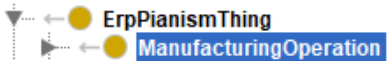


Figure 35 - ManufacturingOperation Class

The **Manufacturing Order** lists the components or parts to be manufactured using a manufacturing resource. This order is issued by the ERP. The order has a unique order code that specifies the quantity and the product to be manufactured. The direct subclass is the ***ManufacturingOperation*** executing the ***ManufacturingOrder***. The ***ManufacturingOrder*** class and subclasses are shown in Figure 36.

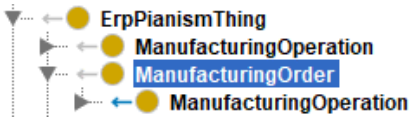


Figure 36 - ManufacturingOrder Class

The **Occurrence** is a maintenance task performed on a resource following an anomalous status of the working conditions. The maintenance task has a request date, an intervention and a resolution time. The ***Occurrence*** class also describes the maintenance executer and the maintenance description. The ***Occurrence*** class is shown in Figure 37.

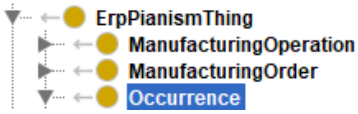


Figure 37 - Occurrence Class

The **Raw Materials** are the base materials in a raw or unprocessed state to be used in a manufacturing order. The direct subclass is ***ManufacturingOrder*** where the raw materials are used. The ***RawMaterials*** class and subclasses are shown in Figure 38.

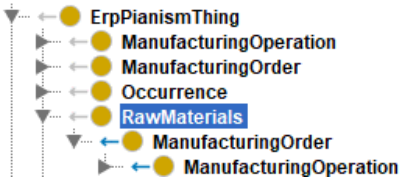


Figure 38 - RawMaterials Class

The **Resource** is a machine, device, equipment, or system able to perform manufacturing tasks. The direct subclass ***Extruder*** represents the definition of an extruding machine. The ***Resource*** class is shown in Figure 39.

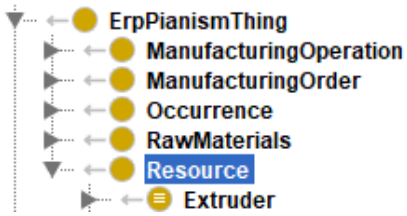


Figure 39 - Resource Class

The **Extruder** in the scope of PIANiSM represents the resource used in the plastic extrusion manufacturing process. This class as a direct ***'is a'*** relation with the class ***Resource***. The ***Extruder*** class is shown in Figure 40 - Extruder Class.

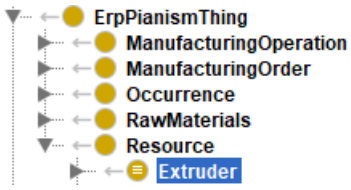


Figure 40 - Extruder Class

8.1.2 Properties

The Ontology Web Language (OWL) has two types of properties:

1. **Object properties** add restrictions and represent relationships between classes or instances. OWL provides a top-level property ***'topObjectProperty'***, and a new sub-property named ***'erpTopObjectProperty'*** was added as a parent for all defined object properties within ***OntoPianismErp***;
2. **Data properties** declare characteristics of instances, such as data types representing, numbers, strings, or dates. OWL provides a top-level property ***'topDataProperty'***, a new sub-property named ***'erpTopDataProperty'*** was added as a parent for all defined object properties within ***OntoPianismErp***.

The properties of ***ErpPianismThing*** subclasses were implemented as follows:

The ***ManufacturingOperation*** class is strongly tied to a Manufacturing Order a by the relation (object property***) 'erpOperationFromOrder'***, knowing that a manufacturing operation is always triggered by a manufacturing order. This class characterizes the details of an order through several data properties using the 'exactly' constraint, namely '**erpArticleCode'**, ***'erpAmmountHarnessed'***, ***'erpJobDescription'***, ***'erpMachineCode'***, ***'erpOrderCode'*** and ***'erpRejectedQuantity'***. Also, due to the temporal nature of the operation, the start and the end

time are specified, the ***'has beginning'*** and ***'has end'*** relationships from ***OWL-Time*** ontology are used to link the class to the ***'Time instant'*** class. The properties of this class are shown in Figure 41.

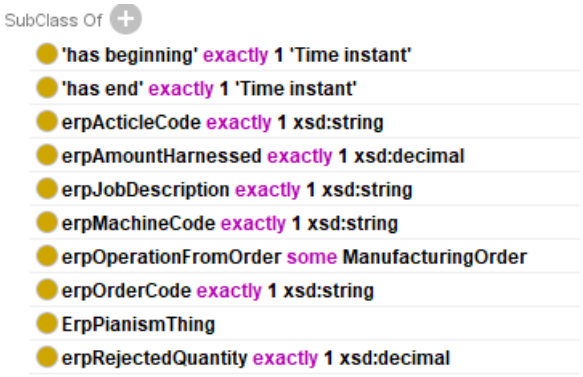


Figure 41 - Manufacturing Operation properties

The ***ManufacturingOrder*** class characterizes the details of an order issued from the ERP, through several data properties using the ***'exactly'*** constraint, namely ***'erpArticleCode'***, ***'erpOrderCode'***, ***'erpQuantity'***, ***'erpSeries'***. The constraint 'some' specifies the use of an undisclosed set of Raw Materials using the object property ***'erpSpecifiesUseOf'***.

The previously described relation between an order and an operation is depicted in Figure 42. The properties of this class are shown in Figure 43.

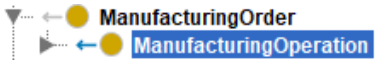


Figure 42 – Relation to Manufacturing Operation

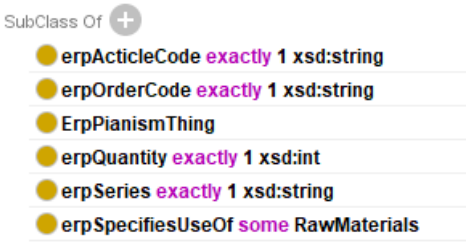


Figure 43 - Manufacturing Order properties

The ***Occurrence*** class characterizes the details of a maintenance occurrence through several data properties using the ***'exactly'*** constraint, namely ***'erpExecuter'***, ***'erPMachineCode',*** ***'erpJobDescription'*** and ***'erpRequestJobDescription'***. Also, due to the temporal nature of the operation, the start and the end time are specified, the ***'has beginning'*** and ***'has end'*** relationships from ***OWL-Time*** ontology are used to link the class to the ***'Time instant'*** class. Furthermore, two additional relationships from OWL-Time ontology are used to link the ***'erpResolutionTime'*** and the ***'erpInterventionTime'*** to the ***'TimeDuration'*** class. The properties of this class are shown in Figure 44.

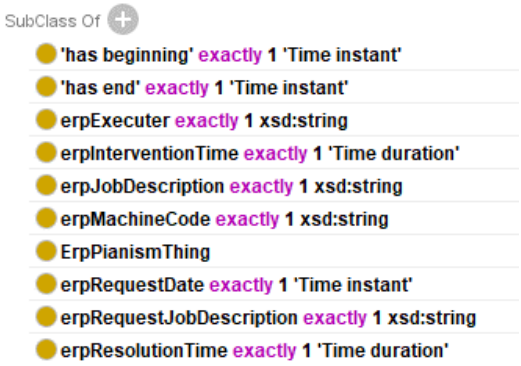


Figure 44 - Occurrence properties

The ***RawMaterials*** class characterizes the raw materials to be used in a Manufacturing Order, having only one data property, ***'erpMaterialCode'*** with the restriction 'some'. The properties of this class are shown in Figure 45. This class is related to a ***ManufacturingOrder***, by the Manufacturing Order's relation ***'erpOperationFromOrder'***, which is evidenced in Figure 46.

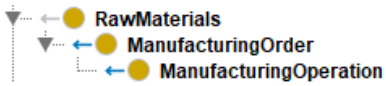


Figure 45 – Raw Materials properties

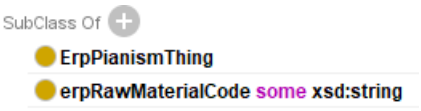


Figure 46 – Relation to Manufacturing Order

The **Resource** has a relation of type ***'is a'*** to the ***OntoPianismERP#Extruder*** class, stating that in this scope a resource is an extrusion machine. The relations of this class are shown in Figure 47.

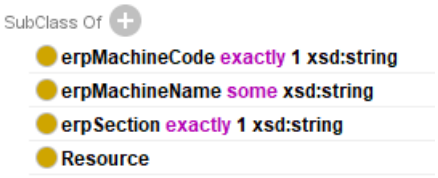


Figure 47 - Resource relation

The **Extruder** represents the physical extruding machine, characterized through several data properties using the ***'exactly'*** or ***'some'*** constraint, namely ***'erpMachineCode'***, ***'erpMachinename*** and ***'erpSection'***. The properties of this class are shown in Figure 48.

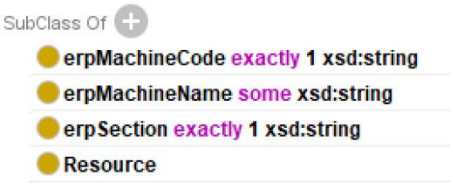


Figure 48 - Extruder Properties

8.1.3 Data and Object Properties

The ***OntoPianismErp*** ontology consists of 15 data properties that are defined under a local property ***'erpTopDataProperty'*** as shown in Figure 50 and consists also of 6 object properties defined under a local object property ***'erpTopObjectProperty'*** as shown in Figure 49. Both local ***'erpTop'*** properties were defined for better readability purposes.

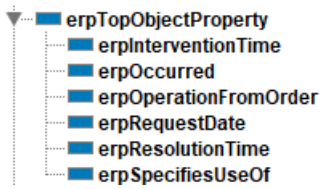


Figure 49 - Object properties

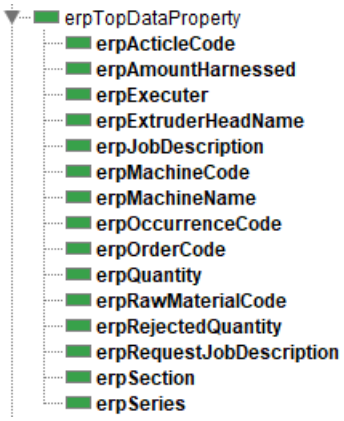


Figure 50 - Data properties

8.1.4 Concepts

The **OntoPianismERP** ontology concepts are presented as classes in Figure 51.

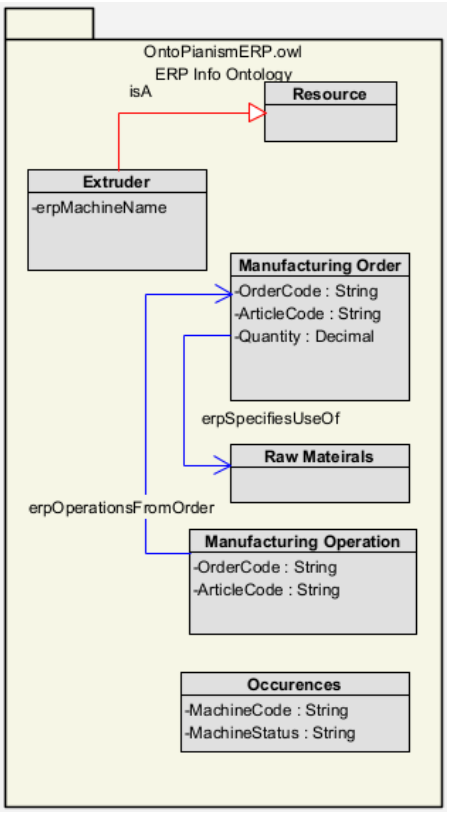


Figure 51 – OntoPianismERP concepts

8.1.5 Relationships

The complete graph of ***OntoPianismErp*** classes and relationships is represented in Figure 52 using the ***OntoGraf*** (Falconer, 2010) Protégé plugin.

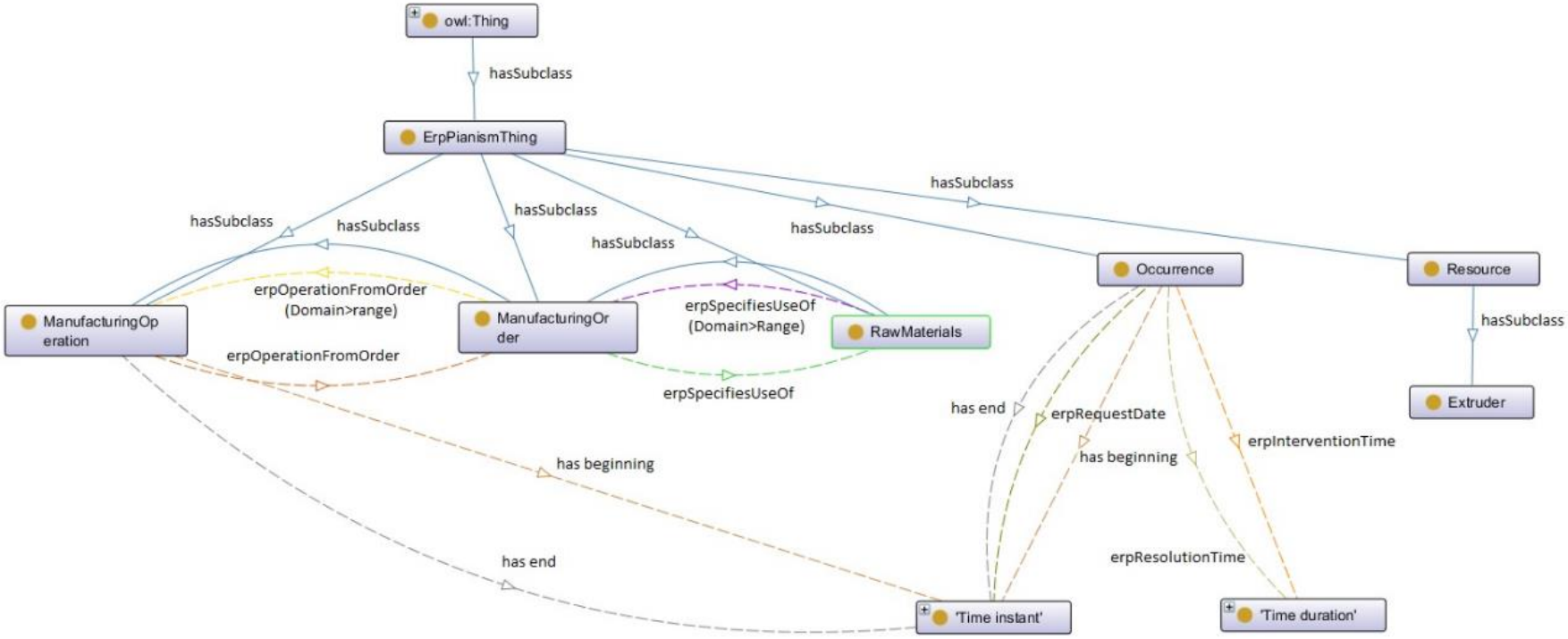


Figure 52 - OntoPianismErp relationships

OntoPianisErp imports the Time Ontology, shown in Figure 53.

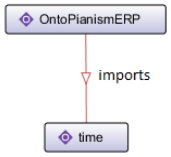


Figure 53 - OntoPianismErp imports

8.2 Domain Ontology Implementation

This section will describe the implementation of the ***OntoProcessMapping*** ontology using Protégé. It will include the description of classes representing concepts, the object properties representing relations, and data properties representing concepts' attributes.

8.2.1 Classes

The ***OntoProcessMapping*** ontology does not have classes of its own. This ontology, being the domain ontology bridging the gap between all the former described, only establishes relations between classes of imported ontologies as represented in Figure 33 of section 7.4.2 Domain Ontology Diagram.

8.2.2 Relationships of imported ontologies

The ontologies imported by ***OntoProcessMapping*** are represented in Figure 54 using the ***OntoViz*** (Michael, 2007) Protégé plugin. The imports represented with green arrows are direct imports, while those represented with grey arrows are indirect imports.

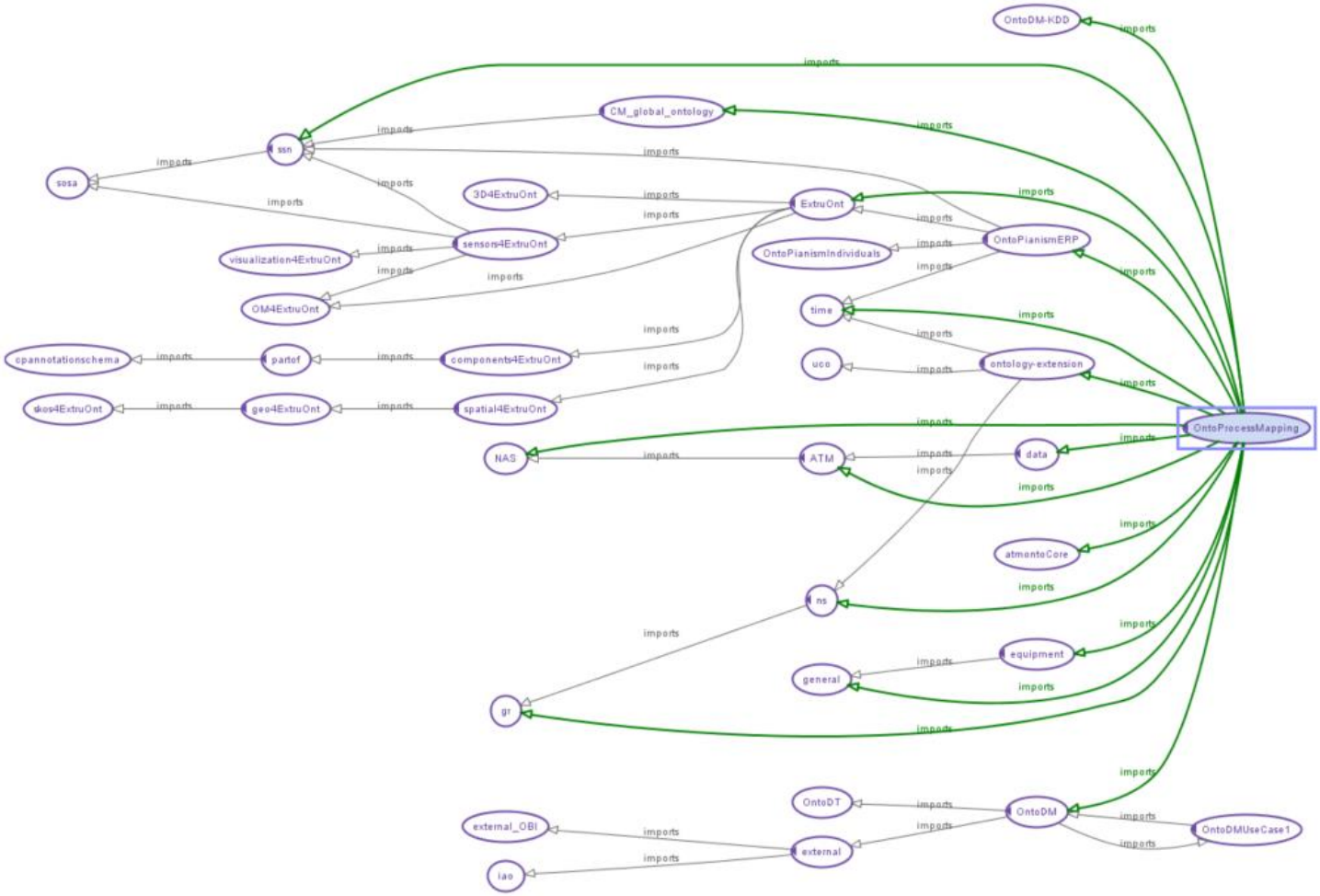


Figure 54 - OntoProcessMapping imports

8.2.3 Properties

The ***OntoProcessMapping*** ontology, not having classes of its own, only adds object properties to the imported ontologies' classes, assuming the role of an aggregation ontology and bridging the gap between all the imported ontologies. This section details for each of the imported ontologies which classes were enriched with additional properties. None of the imported ontologies are modified, as all the additional relations are collected in the ***OntoProcessMapping*** ontology.

8.2.4 Object Properties

The ***OntoProcessMapping*** ontology consists of 7 new object properties defined under a local object property ***'mapTopObjectProperty'*** to group all mapping properties, with the purpose of better readability, as shown in Figure 55.

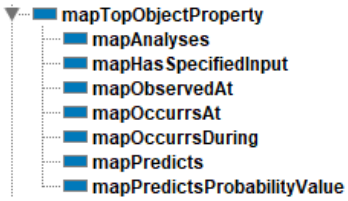


Figure 55 - OntoProcessMapping properties

The new mapping properties are used together with already existing object properties contained in the imported ontologies to perform the relations between the classes of the ontologies encompassing the domain ontology.

8.2.4.1 ASIIO Ontology Properties

The **Event** class has three additional object properties, specifying that an event has an interval during which the event happens, represented by the property ***'interval during'*** of exactly one ***'Date-time interval'*** of class the ***Time*** ontology. Additionally, during an event, some ***'ManufacturingOperation'*** and some ***'Occurrence'*** can occur via the ***'erpOccured'*** object property, both the classes and object properties are relative to the ***'OntoPianismErp'*** ontology. The additional Event properties are represented in Figure 56.

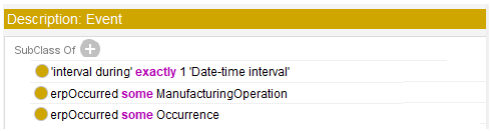


Figure 56 - ASIIO Event properties

8.2.4.2 Time Ontology Properties

The Time Ontology has no properties added to its classes.

8.2.4.3 ExtruOnt Ontology Properties

The **Extruder** class is an equivalent of the ***OntoPianismErp*** ontology **Extruder** class, Figure 57.

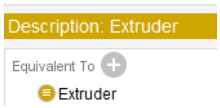


Figure 57 - ExtruOnt Extruder class equivalent

The ***Sensor*** class is an equivalent of the ***SSN*** ontology ***Sensor*** class, shown in Figure 58, and the **Sensor** includes an ***Event*** of the ***ASIIO*** ontology, shown in Figure 59.

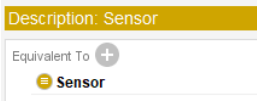


Figure 58 – Sensor equivalent

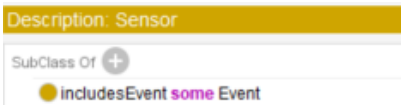


Figure 59 – Sensor includes Event

The ***Extrusion Head*** class is an equivalent of the ***CDM-Core*** ontology ***Component*** class, shown in Figure 60.

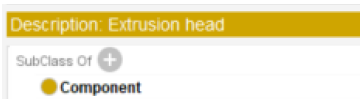


Figure 60 - Extrusion head equivalent

8.2.4.4 SSN Ontology Properties

The ***System*** class identifies that a system has a condition, which might indicate a fault or state, represented by ***'hasSystemCondition'*** of class ***'System_Condition'***, both the data property and class of the ***CDM-Core*** ontology. A ***System*** is composed of several components, which is represented by the ***'hasComponent'*** object property of class ***Component***, here also, both the data property and class of the ***CDM-Core*** ontology System class are shown in Figure 61.

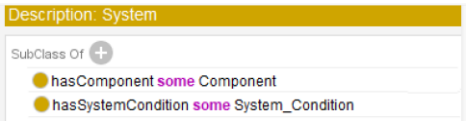


Figure 61 - System Class properties

The ***Property*** class expresses that a sensor property is observed at a specific time instant using the ***'mapObservedAt'*** object property with the cardinality ***'exactly'*** 1 at an ***OWL-Time 'Time Instant',*** Figure 62**.**

Figure 62 - Property observed at Time Instant

8.2.4.5 CDM-Core Ontology Properties

A component condition is indicated by the occurrence of a fault, this condition is represented by the ***Component_Condition*** class and occurs during a time interval, here represented by using the ***'mapOccursDuring'*** object property with the cardinality ***'exactly'*** 1 at an ***OWL-Time 'Time Interval',*** Figure 63**.**

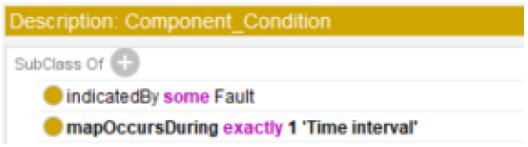


Figure 63 – Component Condition class properties

A fault is indicated by a component condition, represented by the ***Fault*** class, a fault is the result of a symptom and occurs at a specific time and instant using the ***'mapOccuresAt'*** object property with the cardinality ***'exactly'*** 1 at an ***OWL-Time 'Time Instant',***

The ***Fault_State*** class properties are represented in Figure 64**.**

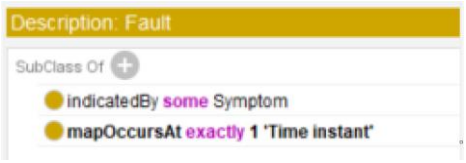


Figure 64 - Fault class properties

A fault state is the result of the occurrence of a fault, represented by the ***Fault_State*** class, this fault state occurs during a time interval, here represented by using the ***'mapOccursDuring'*** object property with the cardinality ***'exactly'*** 1 at an ***OWL-Time 'Time Interval'*.** Simultaneously a fault state represents an event occurring in a component and thus has a direct **'is a'** relation with the ***Event*** class from the ***ASIIO*** ontology. The ***Fault_State*** class properties are represented in Figure 65**.**

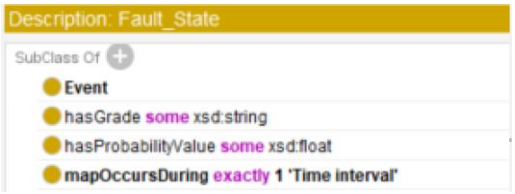


Figure 65 - Fault_State class properties

8.2.4.6 OntoPianismERP Ontology Properties

In the scope of the **OntoPianismErp** ontology, an Extruder represents a Resource used in the plastic extrusion manufacturing process. This resource is a unit of abstraction for pieces of another system which have components and subsystems, which are by themselves other systems. The ***Resource*** class is equivalent to the ***SSN*** ontology ***System*** class which specifies a system as a unit of abstraction for pieces of infrastructure. The Resource class is represented in Figure 66.

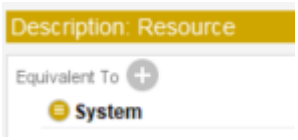


Figure 66 - System class properties

During the manufacturing process the Extruder executing the manufacturing task has a set of temporal events happening, thus the Extruder class has the property ***'includesEvent'*** with constraint ***'some'*** to assign the ***Event*** class from the ASIIO ontology. This property introduces the temporal factor in the events happening in an extruder, knowing that an ***Event 'occurs'*** during a ***Time Interval***. An extruder has a series of Extrusion Heads to perform the extrusion process, this is explicit in the ***'has head'*** object property with cardinality ***'some'*** of ***ExtruOnt*** Ontology ***'Extrusion_head'*** class.

Likewise, an extruder has a variety of sensors described in *1.1.2 Sensor Types*, these sensors are mapped by the ***'has Sensor'*** data property with cardinality '***some'*** of ***Sensor*** Ontology ***'Sensor'*** class. Properties of the Extruder class are shown in Figure 67.

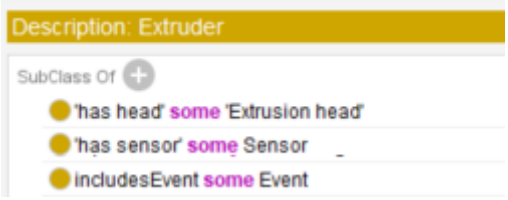


Figure 67 - Extruder class properties

The occurrences happening in the extrusion processes represented by the ***Occurrences*** class, such as breakdowns or maintenance tasks have several temporal properties to characterize the occurrence event. These properties are the request date, the intervention time and the resolution time. These properties are represented by '***erpInterventionTime'*** with a cardinality of '***exactly'*** 1 ***OWL-Time 'Time duration',*** the ***'erpRequestDate'*** with cardinality of '***exactly'*** 1 ***OWL-Time 'Time instant',*** *and the* ***'erpResolutioTime'*** *with* '***exactly'*** 1 ***OWL-Time 'Time duration'.*** The ***Occurrence*** class properties are represented in Figure 68.

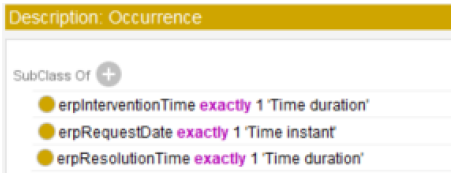


Figure 68 - Occurrence class properties

8.2.4.7 Onto-DM Ontology Properties

To apply ML and DM algorithms some classes of the included ***Onto-DM*** ontology have also data properties added to collect data from the extrusion process and thus establish patterns that could lead to fault prediction. The ***Onto-DM*** ontology ***'algorithm_execution'*** class has property ***'mapAnalyses'*** with cardinality ***'some'*** of class ***'Property'*** of the ***SSN*** Ontology.

The former class has two additional relations with data properties of type ***'mapHasSpecifiedInput'*** with cardinality ***'some'*** to classes ***'Symptom'*** and ***'Component_Condition'***, of the ***CDM-Core*** Ontology.

One important factor to consider, to establish patterns leading to faults are the type of raw materials that are processed in relation to the sensor reading, so a data properties of type ***'mapHasSpecifiedInput'*** with cardinality ***'some'*** creates a relation to the class ***'RawMaterials'*** of the ***OntoPianismErp*** Ontology.

The full list of added properties to the class ***'algorithm_execution'*** is shown in Figure 69.

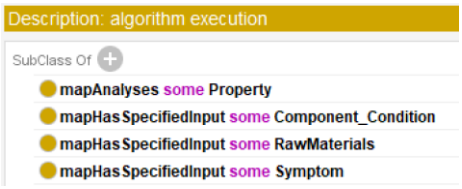


Figure 69 - Onto-DM class properties

The ***'predictive modelling algorithm execution'*** class, defines the execution prediction algorithms Property ***'mapAnalyses'*** with cardinality ***'some'*** of class ***'Property'*** of the ***SSN*** Ontology. The former class has two additional relations with data properties, one of type ***'mapPredictsProbabilityValue'*** with cardinality ***'some'*** to the class ***'Fault_State'*** and a second property ***'mapPredicts'*** to class 'Fault'***,*** both belonging to the ***CDM-Core*** Ontology. The property list is shown in Figure 70.

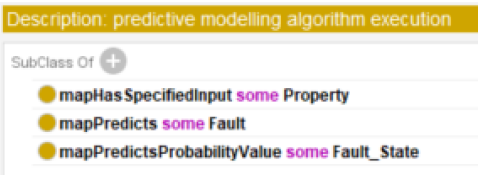


Figure 70 - predictive modelling algorithm execution class properties

9 JSON to Ontology Mapping tool

This chapter details the architecture and the development of a software tool, making a functional application of the previous developed domain ontology ***OntoProcessMapping***, to represent temporally and semantically the raw data collected from a plastic extrusion process.

The tool was named *JSON to Ontology Instance Mapper* (J2OIM) and performs the transformation of time-sensitive data – real-world, time-sensitive sensor data collected from industrial equipment and contextual information from existing management software – into a semantic representation that accounts for its temporally dynamic nature.

The J2OIM tool still as a proof-of-concept was presented in the scientific work "***Applying time-constraints using ontologies to sensor data for predictive maintenance***" (Nobre, Canito, Neves, Corchado, & Marreiros, 2022), and published in "Information Systems and Technologies: WorldCIST 2022", at the "WorldCist'22 - 10th World Conference on Information Systems and Technologies".

9.1 Temporal representation of time-sensitive data

The use-cases described in chapter **6 Solution Design** explain the acquisition of real-time raw data from industrial equipment, this data should be processed according to the temporal representation and constraints defined by the developed domain ontology into semantic data. Once the data is properly transformed it can be used in a PdM process by applying ML and DM algorithms. Nevertheless, the collection of raw data from industrial equipment has not been the subject of extensive in-depth study regarding the temporal nature of the data, especially when considering huge amounts of data provided by continuous streams (Canito, Corchado, & Marreiros, A systematic review on time-constrained ontology evolution in predictive maintenance, 2021).

The transformation of temporal data from diversified sources into semantic structured data is a complex task, and no industry standards are defined in terms of a consensual process or

format. Some tools already exist to perform the transformation of data structures according to a format defined in an ontology, but each tool is tailored for a very specific purpose.

A tool for the semantic representation of data is proposed in **JsonToOnto: Building Owl2 Ontologies from Json Documents** (Sbai, Louhdi, Behja, & Chakhmoune, 2019) by applying a set of transformations to convert a single JSON document into a OWL2 Ontology, not having the capability to relate data from several files and end establish relationships creating new knowledge. Also, no transformation rules can be specified, relying the transformation process only on the JSON's structure, being the rules inferred by the key labels, and the nesting level of the document. This approach does not contemplate configurations or templates, and the structure of the resulting ontology is always based on the structure provided by the input JSON file.

In **Translating JSON Schema logics into OWL axioms** (Cheong, 2019), a tool to transform JSON documents into a predefined ontology structure in terms of OWL classes, objects and data properties was presented. To achieve this transformation, an additional JSON Schema is used as a template to map the original JSON properties to existing OWL classes, objects, and data properties. Nevertheless, the JSON Schema mapping only works with a single JSON document and does not allow the interrelation of several JSON document sources into the same ontology.

The challenge to overcome with the development of the ***J2OIM*** tool is to implement a process that can be systematized and be an enabler in synchronizing several data sources and integrating this data into a unified semantic model. Considering the domain ontology, the representation of temporal entities describing changes in classes and properties over time is already considered by the introduction of the ***ASIIO*** ontology ***Event*** class, specifying that the event happens at an ***'interval during'*** of exactly one ***'Date-time interval'***. This assures that any type of data may have different values at different time points.

The transformation of raw data into semantic data has also to account for the temporal transformation of the properties, and these must preserve the original values even after the temporal transformation. One of the works studied in the State-of-the art showed that the CHRONOS (Preventis, Marki, Petrakis, & Batsakis, 2012), a Protégé (Stanford Center for Biomedical Informatics Research, 2021) plug-in, manages to apply temporal relations to static entities, obtaining as end-result dynamic temporal entities. Regrettably, the Chronos plug-in is limited to its usage within the boundaries of Protégé and thus not suited to be integrated into a process to perform massive and automated transformations.

And so, the decision was made to conceive an entirely new temporal transformation process to be applied to static data originating from manufacturing processes. The approach followed in terms of data modelling was aligned with CHRONOS and the corresponding reasoning tool (Anagnostopoulos, Batsakis, & Petrakis, 2013).

The literature review performed in the State-of-the-art demonstrated that the 4d fluents pattern can represent the persistence of objects through time (Burek, Scherf, & Herre, 2019) by

reifying a temporal part of the object. The developed domain ontology's design accounts for the changes in object qualities over time, by applying the analysed pattern, making the developed data model suitable to collect time-sensitive data from sensors. As thoroughly studied, sensors data changes at each period (t1, tn) and over these periods the quality of data changes in accordance (Burek, Scherf, & Herre, 2019) (Canito, Corchado, & Marreiros, 2021).

Figure 71 exemplifies the conversion of a static relation of the entities ***Machine*** and ***Manufacturing Operation***, by introducing the temporal relation ***Event1*** representing the ***Timeinterval1***, with the ***StartInstant*** and ***EndInstant*** during which the ***Event1*** occurred.

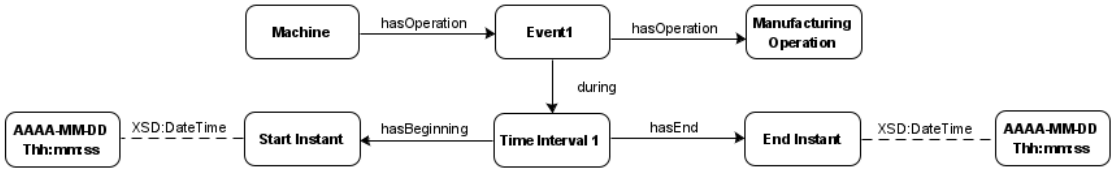


Figure 71 - Temporal object property between entities (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

9.2 The J2OIM Tool Architecture

The J2OIM was developed to have the capability to apply transformations to the raw data acquired from the different heterogeneous sources into instances represented through ontologies. Having as result the semantic representation of data that can be used for reasoning processes to support PdM.

The architecture of the application showing the data flow is depicted in Figure 72. Two main sources of data are considered for this scenario, namely:

- the plastic extruder, which provides data from the extrusion equipment sensors;
- the Enterprise Resource Management (ERP) software, which provides data that describes equipment, manufacturing orders and occurrences, among others.

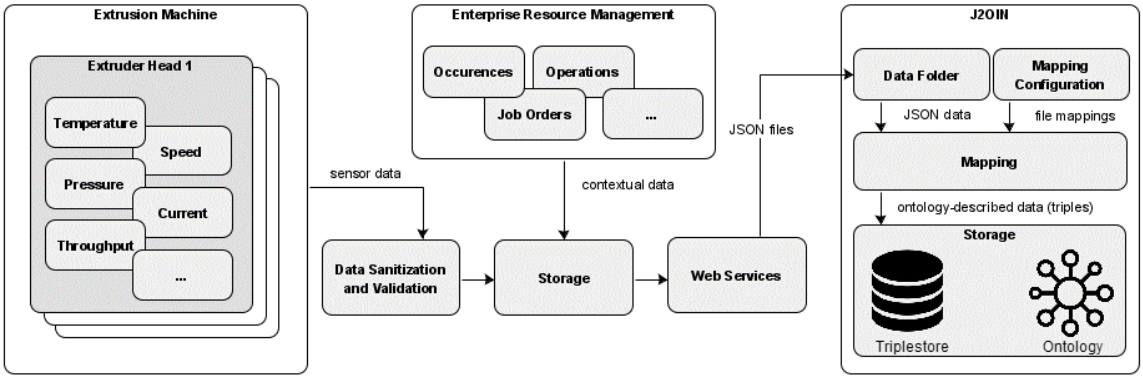


Figure 72 - Data flow, from the individual sensor and software sources to the triple store (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

The conceived architecture accounts for these possibilities by first storing this data into different data files, which are then processed individually as they are semantically similar in terms of the JSON format.

In Table 19, two of such files are presented, with data collected from the endpoints providing data from an extrusion equipment relative to the Machine description and Feed Rate sensor:

Table 19 - Data Samples (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

Machine description data	Feed Rate data

[{
"MachineCode": "VK07",
"MachineDescription":"Extruder7 Layers",
"Section": "S01",
"Extrusion_heads": "A,B,C,D,E,F,G"
}]
	[{
	"timestamp": "2021-09-10T00:00:00.000Z",
	"rawData": "1.787231"
	},
	{
	"timestamp": "2021-09-10T00:01:00.000Z",
	"rawData": "1.787231"
	}]

The complete extent of data files contained 5 files with process data, such as orders, operations occurrences, machine and sensor description, and files representing data from 15 different sensors. All the loosely obtained data, considered for this study, must be related, and transformed into a concise model.

The data samples taken in account in the present work, originated from 3 extrusion machines, with 24 variables representing distinct values, describing the machine, sensors, occurrences, and manufacturing operations. Table 20 categorizes the types of data and total records found in the samples.

Table 20 - Total records of sample data (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

Data origin	File Type	Records
Enterprise Resource Planning data	Machines	3
	Sensor Types	15
	Occurrences	17
	Manufacturing Operations	9.096
	Manufacturing Orders	3.157
Sensor data	Sensor readings	1.001.965
Total		**1.014.253**

Knowing that the obtained data is not normalized and hasn't any relationship, rendering the interpretation difficult, this problem is coped with a dedicated mapping file, conceived in JSON format, to map each property of each file to a triple subject-predicate-object in the domain ontology. The correct mapping of a property enables the semantic transformation into an individual with object and data properties, allowing the once semantic represented data to be inserted into:

- a file with an OWL2 structure, although this file is not an ontology, because it shall contain only individuals, for the sake of simplifying the concept, this file shall be referred to as an **Ontology File**,
- a triple store, in this case, an Apache Fuseki Triple Store.

An example of a mapping configuration definition can be seen in Table 21, which displays the following JSON properties:

- **file**: text, specifies the JSON file from which the information is loaded;
- **purpose**: the literal identification of the entity, much in the sense of an ID or enumeration;
- **namespace**: the owl:Class namespace the instance shall belong to, after the transformation is complete;

- **label**: used to name the individual (or triple), acting as a prefix for the rdfs:label, as each individual triple will contain a universally unique identifier (UUID) suffix generated based on unique data values, with format ***label_UUID***;
- **properties:** a list of mappings of the keys of the JSON file to the object and datatype properties of the instances, including:
    - **jsonProperty:** the property containing the data in the JSON file;
    - **subjectURI:** the prefix of the subject's URI, which is suffixed by a UUID (the same UUID suffixing the label) to form the complete URI, with format ***subjectURI_label*;**
    - **predicate:** the URI predicate for the object or datatype property specified by the jsonProperty.

Table 21 - Mapping configurations snippet (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

Machine data configuration

```
{
   "file": "get Machines.json",
   "purpose": "machines",
   "nameSpace":
"http://www.example.org/Pianism/OntoPianismERP.owl#Extruder",
   "label": "erpExtruder",
   "properties": [
      {
      "jsonProperty": "MachineCode",
      "subjectUri":
"http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpExtruder",
      "predicate":
"http://www.example.org/Pianism/OntoPianismERP.owl#erpMachineCode"
      },
      {
      "jsonProperty": "MachineDescription",
      "subjectUri":
"http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpExtruder",
      "predicate":
"http://www.example.org/Pianism/OntoPianismERP.owl#erpMachineName"
      }]
}
```

The J2OIM tool has three primary modules to execute the transformation processes:

1. the Main module, which orchestrates the transformation process;
2. the Data Loader module, which loads the data according to the defined Model;
3. the Data Writer module, which persists the data as defined in the mapping configuration.

The complete architecture of the JSOIM Tool is shown in Figure 73, the Main module is responsible to launch the J2OIM components and orchestrating the whole process.

Firstly, a controller is launched to execute the loading process of the configuration and data files based on the defined models. Secondly, the loaded data is forwarded via a Batch Loader to an Observable interface. Thirdly, the Data Writer module has observers for the Ontology File Writer and Triple Store Writer. These observers are actively observing the Batch Loader, and for each record passed to the Observable interface, the writers are triggered and persist the data as defined in the mapping configuration file.

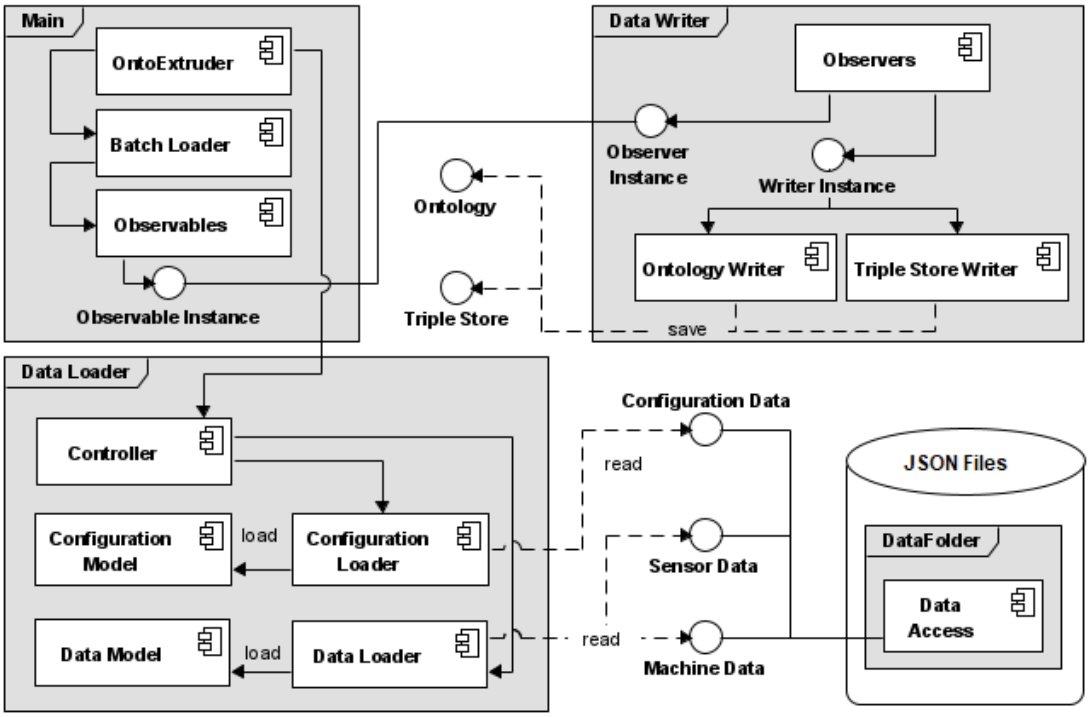


Figure 73 - Component Diagram of the proposed architecture (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

9.3 The J2OIM Tool Implementation

The data transformation process followed several guidelines to correctly represent the incoming data into a valid semantic model. The captured data had to be tagged or classified with an identifier or reference to facilitate establishing relationships using object or datatype properties. For this implementation, each data record read from the source JSON files was tagged with a UUID generated in runtime, based on unique data values during the transformation phase. To assure that the generation of the UUID is idempotent, the unique data values used to generate the UUID were based on timestamps, order codes, or any other data unique to each data record. To provide readable information regarding each individual or subject created in the Ontology File and Triple Store, each subject reference is composed of the

subjectUri defined in the mapping configuration, suffixed with the ***UUID*** based on its data contents. This ensures that any subject, even if inserted or referenced multiple times, always has the same reference, assuring the idempotency of the process.

The main purpose of this implementation is the representation of time-sensitive data related to extruder machines, supplied by sensors, in temporal N-ary relations, whenever applicable. However, not all subjects detected in the obtained data refer to time-sensitive data; some subjects have simpler non-temporal relations with other objects.

The discovered temporal relations associated with the extruder machine were: manufacturing operations and manufacturing occurrences originating from the ERP data, and the temporal relations associated with the sensors, where all the sensed output values occurred over time.

For each identified temporal relation, a new ***Event*** instance is created and then linked to the original instance: the ***Event*** creates a new relationship between two instances that occur within a specific Time Interval.

Table 22 presents an overview of all the individuals or subjects, in form of triples, distinguishing those identified as non-temporal and temporal relations.

Table 22 - Relations between subjects (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

Subject	Predicate	Object	Relation type
Machine	hashead	Head	non-temporal
Head	hassensor	Sensor	non-temporal
Sensor	includes event	Event	temporal
Manufacturing Operation	from order	Manufacturing Order	non-temporal
Machine	includes event	Event	temporal
Event	hasOutput	Value	temporal
Event	executed	Manufacturing Operation	temporal
Event	occurrence	Manufacturing Occurrence	temporal
Event	during	Time Interval	temporal
Time Interval	hasBeginning	Time Instant	temporal
Time Interval	hasEnd	Time Instant	temporal
Time Interval	**hasEnd**	**Time Instant**	**temporal**

To exemplify how an initial non-temporal relation such as the ***Sensor-hasOutput-Value*** is transformed, the same relation is now represented as a temporal relation:

- Sensor-includesEvent-Event;
- Event-hasOutput-Value;
- Event-during-TimeInterval.

Figure 74 shows how several single JSON file entries are converted into the instances required by the Domain Ontology. The machine is now related to the sensors present in the machine head and the sensor data is related to the precise time interval in which the reading occurred.

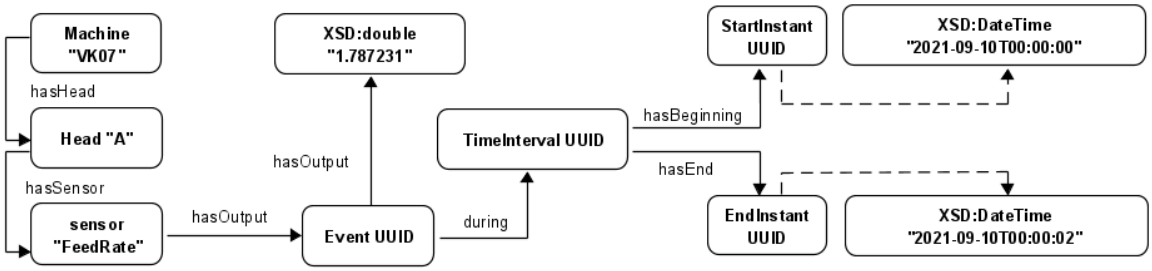


Figure 74 - Events occurring in sensors (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

Figure 75 shows how numerous and more complex actions observed on a particular machine are now all interrelated and temporally represented. Here, a machine executes manufacturing operations which fullfil manufacturing orders and on the same machine occurrences are associated with the interval in which they occur.

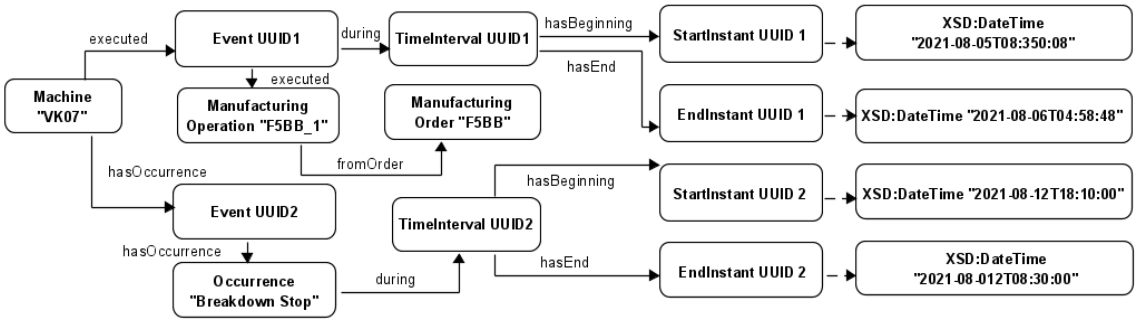


Figure 75 - Events occurring in machines (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

The individuals persisted in the Ontology File, such as the extrusion head sensors instances, have now temporal relations to represent the output values occurring over time.

Using the ***OWL Turtle Syntax*** for the sake of representation simplicity, in Table 23 an individual representing an extrusion head sensor has 3 temporal events, each one of these events represents a time interval and an associated value read by the sensor “FeedRate” of head “A” in the extruder machine “VK07”.

The time interval is comprehended between two moments, ***hasBeginning*** and ***hasEnd***, containing the timestamps representing the exact moments in time

Table 23 - Individual of class Extrusion head sensor (FeedRate) and output Event (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

```
OntoPianismIndividuals:erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310 rdf:type
owl:NamedIndividual ,
j.2:Sensor ;
j.4:hasOutput OntoPianismIndividuals:erpEvent_2dedebc6-9e3b-32e9-aa94-49b75c94500d ,
                 OntoPianismIndividuals:erpEvent_c17a2ae3-2b86-3b2c-a0fb-02cc14b95cec ,
                 OntoPianismIndividuals:erpEvent_e51b89a6-8392-3cf2-825d-53bb9d9a8f7e ;
sensors4ExtruOnt:sensorName "FeedRate" ;
CM_ontology:hasSensorID "63b006a9-b414-31d4-a2d4-00ee004fe310" ;
rdfs:label "erpSensor_VK07_A_FeedRate" .

OntoPianismIndividuals:erpEvent_2dedebc6-9e3b-32e9-aa94-49b75c94500d rdf:type
owl:NamedIndividual ,
j.0:extEvent ;
j.3:intervalDuring OntoPianismIndividuals:erpTimeInterval_7b0c803e-c1af-3072-4e694d5b161d;
j.5:hasOutput "1.787231" ;
   rdfs:label "erpEvent_2dedebc6-9e3b-32e9-aa94-49b75c94500d" .

OntoPianismIndividuals:erpTimeInterval_7b0c803e-c1af-3072-4e694d5b161d rdf:type
owl:NamedIndividual ,
  j.3:DateTimeInterval ;
  j.3:hasBeginning OntoPianismIndividuals:erpTimeInstant_c3832c2e-8548-3e9e-8e9d-
1fc057c57fa1 ;
  j.3:hasEnd OntoPianismIndividuals:erpTimeInstant_ab441966-2b58-3aff-b0a8-1d7836e7a485 ;
  rdfs:label "erpTimeInterval_7b0c803e-c1af-3072-b62a-4e694d5b161d" .
```

The resulting instances obtained from the data transformation are listed in Table 24. These show that non-temporal data from the ERP resulted in the same number of instances as the initial data, due to a lack of temporal transformation. However, sensor readings resulted in about half of the initial values, due to eliminations of repeated values over the same time interval. Nonetheless, the creation of temporal instances means that the total number of instances almost doubled, with about a million new instances being created for the temporal representation.

Table 24 - Total resulting instances (Nobre, Canito, Neves, Corchado, & Marreiros, 2022)

Data origin	Instances	Records
Enterprise Resource Planning data	Machines	3
	Sensors in extrusion heads	121
	Occurrences	17
	Manufacturing Operations	9.096
	Manufacturing Orders	3.157
Sensor data	Sensor readings (data properties)	502.607
Temporal data from ERP and sensors	Events	468.228
	Intervals and Time Stamps	500.605
Total		**1.483.834**

9.4 The Semantic transformed data

In terms of more practical usage, after the development of the domain ontology, the development of the ***J2OIM*** tool was of paramount importance. Not only allowing the transformation of the raw data according to the domain but particularly enabling the consequent storage of the transformed data by the ***J2OIM*** tool, was a major achievement to have the data readily available to apply ML and DM algorithms. Among the possible choices, allowing the storage of knowledge in the form of triples (subject-predicate-object), two types of storage were selected:

- **the OWL2 storage,** storing the triples as individuals or instances of the domain ontology classes, in a structured OWL2 file. This file hence in OWL2 format is suitable to be imported or loaded with any tool capable of representing OWL2 files, e.g. the Protégé Tool. This representation has also the advantage of facilitating the import of the file containing the individuals in conjunction with the domain ontology, having so a complete overview of the domain classes and the extracted instances represented by the individuals;

- **the Apache Fuseki Triple Store storage**, a database allowing the storage of large amounts of semantic data, providing the SPARQL 1.1 query language to extract, filter and group the stored triples.

9.4.1 OWL2 File Storage

Once stored as triples in the OWL2 file, the extrusion head sensors instances have temporal relations to represent the output values occurring over time. In Figure 76 an individual representing an extrusion head sensor has 3 temporal events, each one of these events represents a time interval and an associated value read by the sensor “FeedRate” located on head “A” of extruder machine “VK07”.

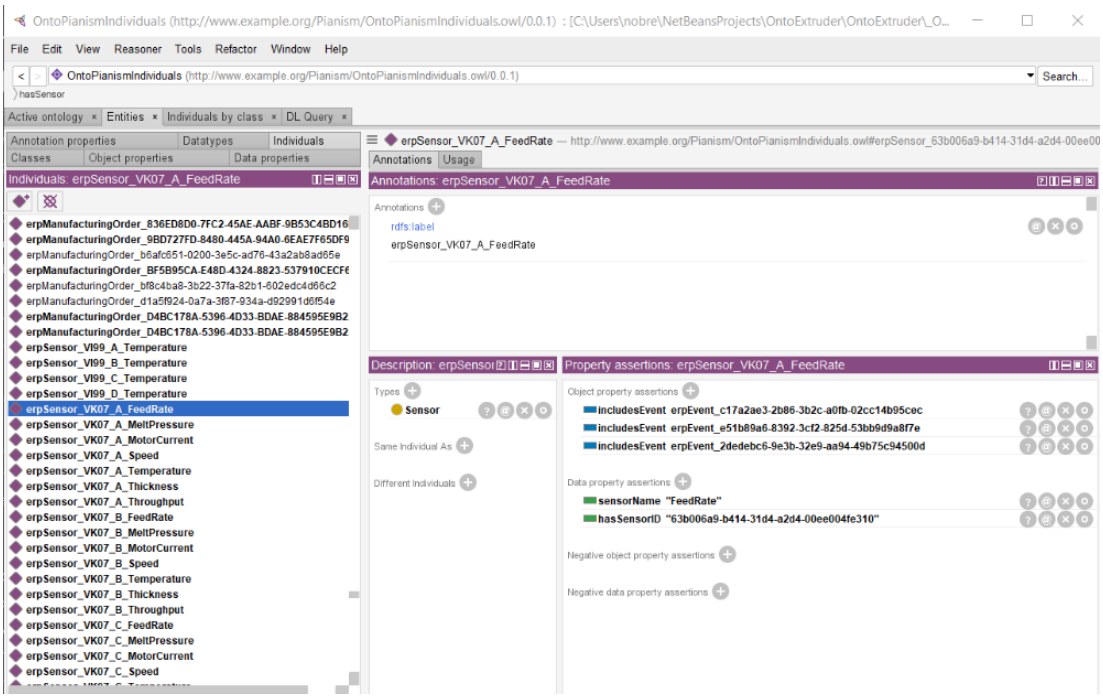


Figure 76 – Extrusion head sensor Individual

9.4.2 Triple Store Storage

Apache Fuseki (Apache Jena, 2021) was used as a standalone server, accessible over HTTP. Based on the same mapping configuration used to persist individuals, the raw data is mapped into triples representing subject-predicate-object.

These triples are then inserted into the triple store using SPARQL’s (W3C, 2021) query and update functions.

The query represented in Figure 77 will show that the same instances that were generated to populate the ontology model have been properly inserted into the Fuseki triple store.

Following the same example as for the OWL2 file, after executing the SPARQL query to return all the triples created for the sensor “Feedrate” located on head “A” on extruder machine “VK07”, which subjectUri “erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310”, the results obtained are equal to those obtained by accessing the ***OntoPianismIndividuals.owl*** ontology file.

```
SELECT ?subject ?predicate ?object

WHERE {VALUES
  ?subject { <http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310>}
  ?subject ?predicate ?object
}
```

Figure 77 - SPARQL query to select a sensor

The detail of the obtained triples corresponding to the sensor “Feedrate” are listed in Figure 78, as for the equivalent individuals, the sensor “Feedrate” located on head “A” of extruder machine “VK07” has 5 triples.

Non-temporal triples represent sensor-specific data and temporal triples represent events.

QUERY RESULTS

Table | Raw Response

Showing 1 to 5 of 5 entries

Search: Show 50 entries

	subject	predicate	object
1	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310>	<http://bdi.si.ehu.es/bdi/ontologies/ExtruOnt /sensors4ExtruOnt#sensorName>	"VK07_A_FeedRate"
2	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310>	<http://www.example.org /CM/CM_ontology.owl#hasSensorID>	"63b006a9-b414-31d4-a2d4-00ee004fe310"
3	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310>	<http://www.loa-cnr.it/ontologies /DUL.owl#includesEvent>	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpEvent_2dedebc6-9e3b-32e9-aa94-49b75c94500d>
4	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310>	<http://www.loa-cnr.it/ontologies /DUL.owl#includesEvent>	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpEvent_e51b89a6-8392-3cf2-825d-53bb9d9a8f7e>
5	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpSensor_63b006a9-b414-31d4-a2d4-00ee004fe310>	<http://www.loa-cnr.it/ontologies /DUL.owl#includesEvent>	<http://www.example.org/Pianism /OntoPianismIndividuals.owl#erpEvent_c17a2ae3-2b86-3b2c-a0fb-02cc14b95cec>

Figure 78 - Feedrate Sensor triples (5 of total 5)

10 Experiences and Evaluation

In this chapter, the details to evaluate the success of the developed ontologies are presented, the research hypotheses, as well as the evaluation and experiments conducted are described in detail.

The primary methodology of the evaluation approach follows a task-based method. This approach was chosen since it measures to which degree the developed ontology improved and facilitated the resolution of the primary goals (Raad & Cruz, 2015).

Secondly, the evaluation includes a set of manual and automated validations to assess the ontology's correctness regarding the requirements.

Thirdly, the publication of a paper relevant to the application of the proposed ontologies, and a journal article exposing further application of the proposed ontologies, however the latter still in submission at the moment of the submission of this dissertation.

Finally, the analysis of sample data collected in real-time from PIANiSM (PIANiSM, 2020) transformed with the ***J2OIM*** tool accordingly to the ontology structure.

10.1 Research Hypothesis

Based on research findings revealed State of the Art, industrial systems have a multitude of different notations to represent the system itself and the sources of collected data. This question is fundamental to be answered so that knowledge can be shared with other systems without the need for extensive mapping and data transformation between these different systems.

The raised questions lead to the formulation of the following hypotheses:

1. How to clearly define a terminology for the context of time-sensitive data in the scope of plastic extrusion equipment?

   Hypothesis 1: Establish clear definitions and concise terminology for the necessary concepts, creating an unambiguous semantic layer to represent the data in the ontology;

2. How to achieve the ease of interconnectivity of systems promoting an efficient exchange of knowledge?

   Hypothesis 2: Defining concise terminology in an integrated ontology bridging the gap with existing ontologies, representing the technical requirements;

3. How to re-use the existing ontologies to extract additional knowledge not yet defined nor correctly interconnected?

   Hypothesis 3: Defining an ontology with a broader scope by integrating and relating the existing ontologies to provide a wide field of application and representation of the data collected from the involved systems;

4. How to represent time-sensitive data semantically accounting for its temporal nature??

   Hypothesis 4: Defining an ontology by applying a 4d fluents pattern representing the persistence of objects through time.

10.2 Information Sources

The indicators applied to support the ontology evaluation are the following:

1. Range of distinct sensor data sources that can be collected and linked together using the ontology;
2. Correctness and exactness in describing the required scenarios in the solicited use cases;
3. Proficiency in combining and representing data collected from real-time sources;
4. Obtaining the validation of the ontology by partners and involved entities;
5. Scientific community, by the publication of a paper expressing the application of such an ontology.

Information acquisition from distinct sources:

1. **Publicly available sources:** online publications, papers covering the scope of real-time representation of data using an ontology, online available ontologies covering the scope of manufacturing processes;
2. **Private accessible sources:** the partners' and entities' contribution to the development of the ontology and the impact on their projects, confirming the approval of the proposed ontologies.

10.3 Evaluation Methodology

Once the ontology reached a stable version considering all the requirements the following evaluations are ready to be executed:

To assess **Hypothesis 1**, the verification if the scope of the proposed ontology covers all the concepts present in the data collected from PIANiSM (PIANiSM, 2020) Web Service, must be accomplished. This verification intends to ensure that the proposed ontology's concepts and relationships fully cover the scope of the manufacturing process.

To prove **Hypothesis 2**, evaluating the efficiency of the developed ontology would entail assessing the capacity to represent data collected from different systems – the data from the ERP and that from the extrusion equipment – and the capacity to represent it using temporal relations. This hypothesis is validated by a manual and automated validation of the ontology structure, and through the scientific community by the means of the publication of a scientific paper exposing the ontology and its the field of application.

To prove **Hypothesis 3,** the possibility of detecting new patterns from the systems' available data should be possible. An example would be patterns that can be extracted from extrusion equipment sensor data and the ERP manufacturing process records. Typically, data from these systems is not related. However, using the proposed ontology should provide means to extract new knowledge once the relationships between data are made explicit. This knowledge is not possible to obtain when data is observed separately. The defined ontology should be capable to represent and correlate all data gathered from an extrusion manufacturing process.

To prove **Hypothesis 4,** a tool was developed to enable the transformation of unstructured data as Individuals or Objects according to the proposed ontology. This tool, **JSON to Ontology Instance Mapper (*J2OIM*)** (Nobre, Canito, Neves, Corchado, & Marreiros, 2022), uses a JSON structure to map fields from the sample data to the proposed ontology's classes and uses object and data properties to establish relationships between previously unrelated concepts, also enabling the representation of time-sensitive data temporally.

10.4 Evaluation Accomplishment

This section presents the validation processes made to ensure the validity of the proposed ontology. These processes targeted OWL validation in terms of structure and syntax, the analysis of sample data, the creation of instances (individuals) according to the ontologies' structure, and the validation by the partners. The evaluations targeted the two developed ontologies:

1. ***OntoProcessMapping***, the domain ontology gathering all auxiliary ontologies, by direct imports;

2. ***OntoPianismERP***, the auxiliary ontology describing data collected from PIANiSM (PIANiSM, 2020).

10.4.1 Verification of Scope

The data obtained from the PIANiSM (PIANiSM, 2020) through several endpoints of Web Service covers the following categories of data from an extrusion manufacturing process:

- Machines
- Sensors
- Job Orders
- Occurrences
- Operations
- Layer Percentage
- Hose Percentage
- Instant Weight
- Pressure
- Motor Rotation
- Fusion Temperature
- Job Unity
- Job Velocity
- Melt Pressure
- Speed
- Motor Current
- Throughput
- Feed Rate
- Thickness
- Temperature

A careful verification of the data must be performed to assure the proposed ontology contains entities to represent all the categories of collected data and has the necessary object and data properties to interrelate the data semantically and temporally.

10.4.2 Manual Evaluation

To execute a systematic validation of the proposed ontology, the guidelines followed were those defined by OD101 (Noy & McGuinness, 2001). These guidelines assured a meticulous verification in terms of correctness and validity throughout the manual validation process. This process targeted the main proposed ontology ***OntoProcessMapping*** and the specific ontology describing data collected from PIANiSM (PIANiSM, 2020), ***OntoPianismERP***.

Validation of the Class Hierarchy consisted in verifying the transitiveness of the hierarchy by the means of checking all direct relations to and from sub-classes, also the verification of the absence of ambiguities was performed in order not to have classes with similar names representing different concepts. Additionally, tracing was performed to assure the inexistence of circular references in class relations each relation to its sub-classes.

Validation of Disjoint Classes, by checking that all classes at the same hierarchical level do not overlap and are explicitly declared as disjointed from one another. This validation was performed to assure that an instance or individual can not belong to two or more disjoint classes. Each class was explicitly declared disjoint to avoid future overlapping.

Adopting Naming Conventions, all object and data properties follow the "camelCase" convention and are prefixed by an identifier allowing quick identification of properties of the same genesis. Also, all properties were grouped under a Top Property to easier represent these properties in a tree view in the Protégé Tool. Following this convention, in the main proposed ontology ***OntoProcessMapping,*** the object properties have the prefix ***map***, to identify them as properties mapping concepts from several ontologies and are grouped by a ***mapTopObjectProperty***, as shown in Figure 79.

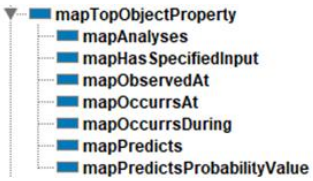


Figure 79 - OntoProcessMapping mapTopObjectProperty

In the specific ontology ***OntoPianismERP***, object properties and data properties have the prefix ***erp***, to identify them as properties mapping concepts from several ontologies and are grouped by an ***erpTopObjectProperty*** and ***erpTopDataProperty***, as shown in Figure 80.

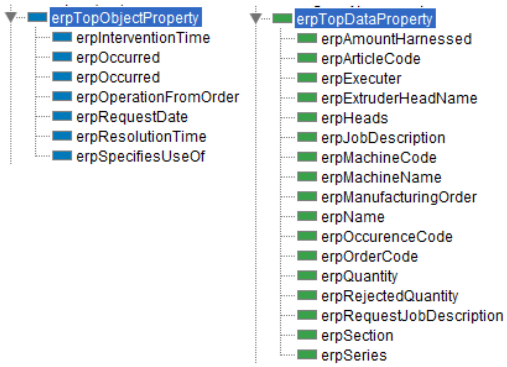


Figure 80 – OntoPianismErp erpTopObjectProperty and erpTopDataProperty

In the specific ontology ***OntoPianismERP,*** the "PascalCase" convention was adopted to create classes representing new concepts. These classes were also grouped by an ***ErpPianismThing*** Class to characterize and differentiate them from all the other classes of imported ontologies, as shown in Figure 81.

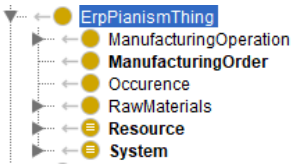


Figure 81 - OntoPianismERP ErpPianismThing

10.4.3 Automated Evaluation

To perform the automated evaluation, a dedicated ontology analysis tool available in the World Wide Web was employed, namely OOPS!, short form of OntOlogy Pitfall Scanner! [8] (Poveda, 2021). As an online tool, OOPS! takes an RDF file as input for the analysis, as shown in Figure 82. The analysis performed by OOPS! verifies the most common ontology pitfalls in ontology development and classifies them as Critical, Important and Minor. Nevertheless, OOPS! uses textual analysis, and thus each identified pitfall should be reviewed regarding its legitimacy. The recommendation of OOPS! is that Critical and Important pitfalls should be fixed. While minor pitfalls are not considered urgent, their resolution contributes to a neater ontology, easier to read and maintain.

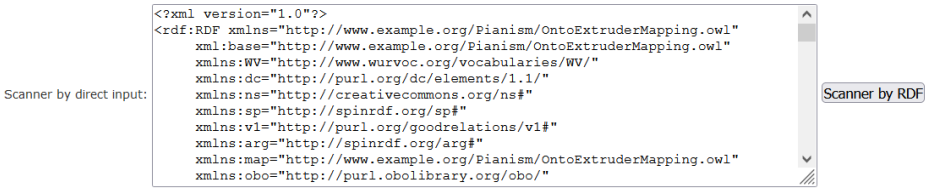


Figure 82 - OOPS! Ontology Scanner

10.4.3.1 Analysis of OntoProcessMapping with OOPS!

Only Minor Pitfalls were identified by OOPS! on the main proposed ontology **OntoProcessMapping**. These were the result of having a set of unconnected ontology elements due to the large set of imported ontologies and do not pose any risk. Minor Pitfall are shown in Figure 83.

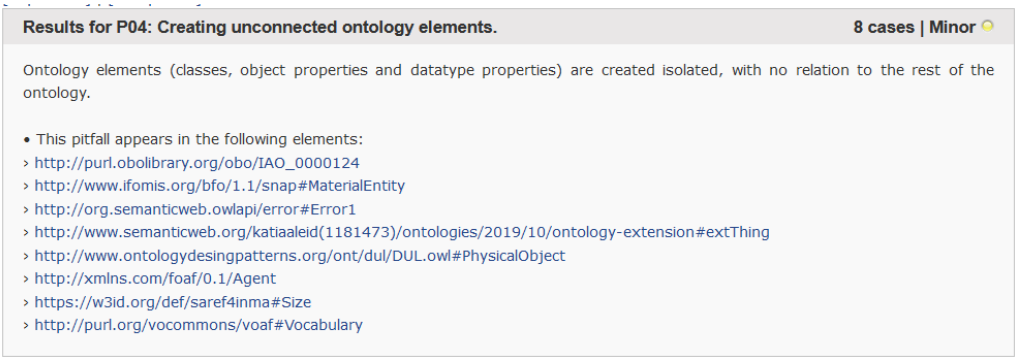


Figure 83 - OntoProcessMapping Minor Pitfalls

[8] OOPS! [Online] Available at http://oops.linkeddata.es/ (Accessed: 31-12-2021)

10.4.3.2 Analysis of OntoPianismERP with OOPS!

Minor and Important pitfalls were identified by OOPS! in the specific ontology ***OntoPianismERP,*** as demonstrated in Figure 84.

Results for P04: Creating unconnected ontology elements.	2 cases | Minor
Results for P08: Missing annotations.	32 cases | Minor
Results for P11: Missing domain or range in properties.	29 cases | Important
Results for P13: Inverse relationships not explicitly declared.	29 cases | Minor
Results for P22: Using different naming conventions in the ontology.	ontology* | Minor
Results for P24: Using recursive definitions.	2 cases | Important
Results for P41: No license declared.	ontology* | Important
SUGGESTION: symmetric or transitive object properties.	16 cases

Figure 84 – OntoPianismERP evaluation results

Important Pitfalls P11 – This pitfall defines that some properties have a "Missing domain or range in properties" flaw, shown in Figure 85. Some of these pitfalls were identified in properties of imported ontologies and therefore left unresolved. The properties belonging to the specific ontology, with the prefix ***erp*** were analysed and a domain and range were added to each of these properties.

Results for P11: Missing domain or range in properties. — **32 cases | Important**

Object and/or datatype properties without domain or range (or none of them) are included in the ontology.

• This pitfall appears in the following elements:
› http://www.w3.org/2006/time#hasDuration
› http://www.w3.org/2006/time#hasTime
› http://www.w3.org/2006/time#hasDurationDescription
› http://www.example.org/Pianism/OntoPianismERP.owl#erpOperationFromOrder
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extTopObjectProperty
› http://www.example.org/Pianism/OntoPianismERP.owl#erpResolutionTime
› http://www.example.org/Pianism/OntoPianismERP.owl#erpInterventionTime
› http://www.example.org/Pianism/OntoPianismERP.owl#erpTopObjectProperty
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extHasStatus
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extStatusOf
› http://www.example.org/Pianism/OntoPianismERP.owl#erpSpecifiesUseOf
› http://www.example.org/Pianism/OntoPianismERP.owl#erpOccurred
› http://www.w3.org/2006/time#day
› http://www.w3.org/2006/time#year
› http://www.w3.org/2006/time#month
› http://www.example.org/Pianism/OntoPianismERP.owl#erpExecuter
› http://www.example.org/Pianism/OntoPianismERP.owl#erpOrderCode
› http://www.example.org/Pianism/OntoPianismERP.owl#erpSection

Figure 85 - OntoPianismERP Pitfalls P11

Important Pitfalls P24 – This pitfall detects "Using recursive definitions" among some properties, shown in Figure 86. Both pitfalls were detected in imported ontologies and therefore left unresolved.

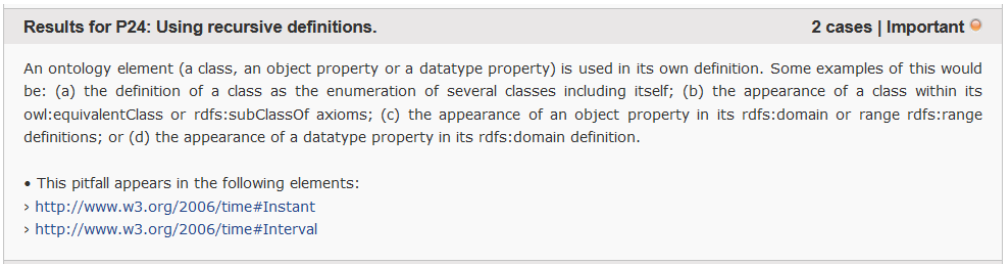


Figure 86 - OntoPianismERP Pitfalls P24

Important Pitfalls P41 – Identifies the ontology as having "No license declared", as shown in Figure 87. This issue was resolved by adding a "dcterms:license" clause to the ontology's definition. The license selected, envisioning the future sharing of this work, but requiring the proper credit, was the Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) (Creative Commons, 2022). The inclusion of this licence assures that other researchers can make use of this ontology.

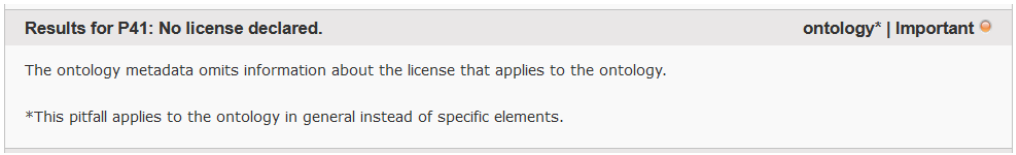


Figure 87 - OntoPianismERP Pitfalls P41

Minor Pitfalls P04 – Informs that an occurrence of "Creating unconnected ontology elements" was detected, as shown in Figure 88. This pitfall happened since the local ***ErpPianismThing*** is considered an isolated element, not having any relationship to other upper-level concepts, this is the consequence of this class being created to concentrate all ERP concepts to be integrated into the ***OntoProcessMapping***. The same pitfall was detected in the imported ontology with the ***extValuepartition*** class. Not being a major flaw, this pitfall was kept unchanged and the ***ErpPianismThing*** left as is.

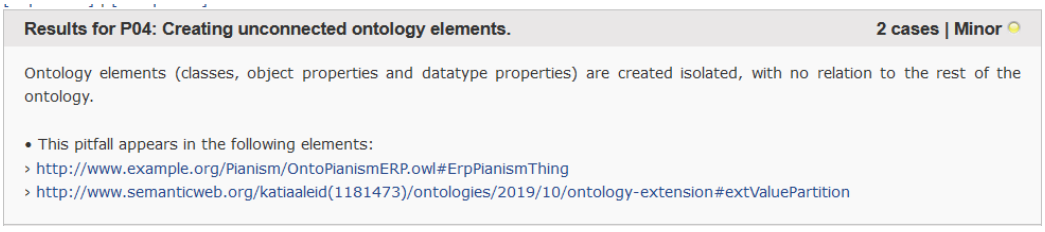
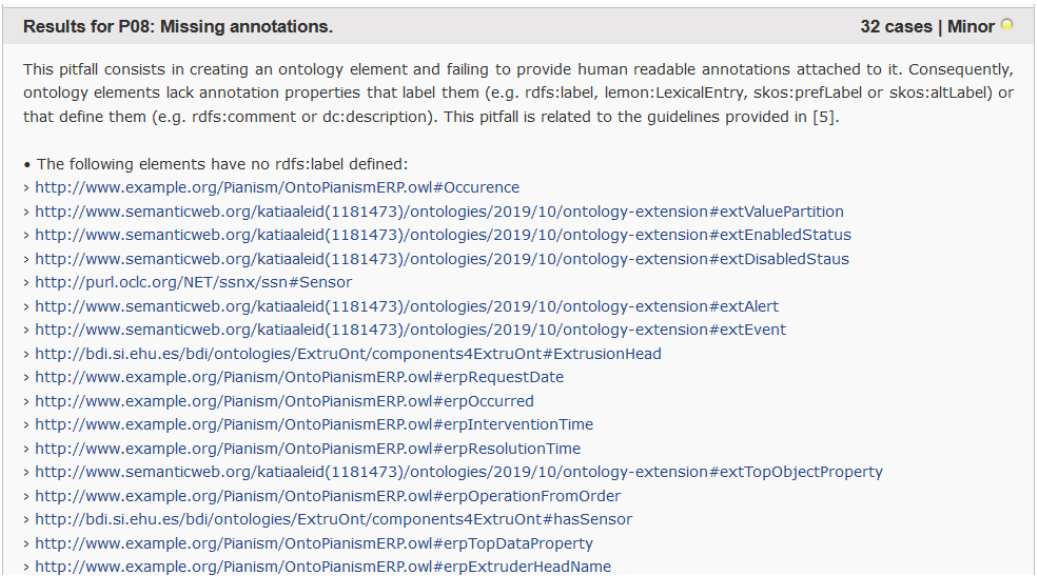


Figure 88 - OntoPianismERP Pitfalls P04

Minor Pitfalls P08 – Identifies **"Missing Annotations"** concerning several entities. This pitfall, although not a contribution to any malfunctioning of the ontology, is an important alert that data properties and object properties should be documented, as shown in Figure 89. As such, besides having already added an **rdfs:label** to some properties to document them in detail, both the **rdfs:label** and a **skos:definition** were added to each property with a concise description of their purpose.



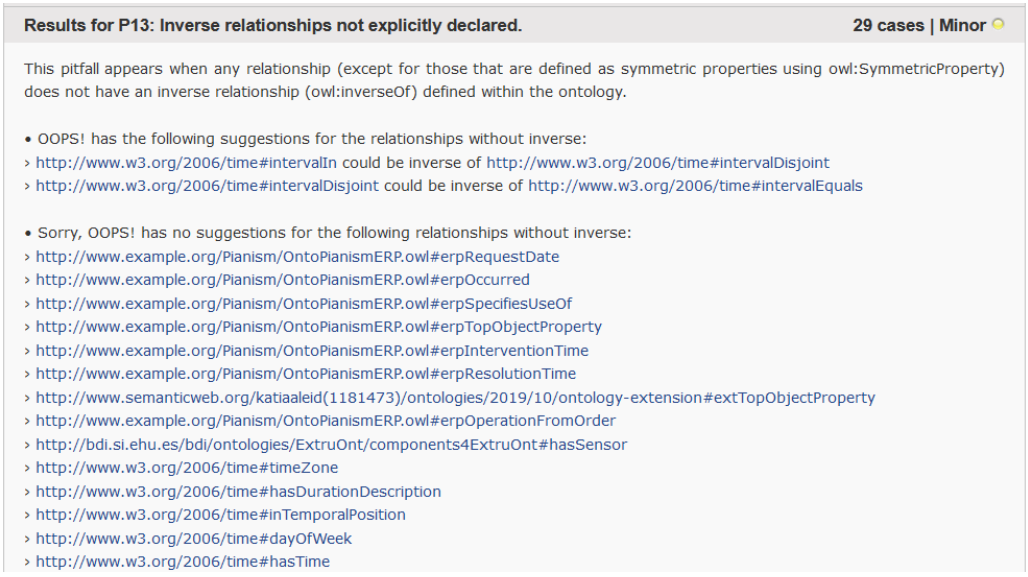
Results for P08: Missing annotations. **32 cases | Minor**

This pitfall consists in creating an ontology element and failing to provide human readable annotations attached to it. Consequently, ontology elements lack annotation properties that label them (e.g. rdfs:label, lemon:LexicalEntry, skos:prefLabel or skos:altLabel) or that define them (e.g. rdfs:comment or dc:description). This pitfall is related to the guidelines provided in [5].

• The following elements have no rdfs:label defined:
› http://www.example.org/Pianism/OntoPianismERP.owl#Occurence
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extValuePartition
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extEnabledStatus
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extDisabledStaus
› http://purl.oclc.org/NET/ssnx/ssn#Sensor
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extAlert
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extEvent
› http://bdi.si.ehu.es/bdi/ontologies/ExtruOnt/components4ExtruOnt#ExtrusionHead
› http://www.example.org/Pianism/OntoPianismERP.owl#erpRequestDate
› http://www.example.org/Pianism/OntoPianismERP.owl#erpOccurred
› http://www.example.org/Pianism/OntoPianismERP.owl#erpInterventionTime
› http://www.example.org/Pianism/OntoPianismERP.owl#erpResolutionTime
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extTopObjectProperty
› http://www.example.org/Pianism/OntoPianismERP.owl#erpOperationFromOrder
› http://bdi.si.ehu.es/bdi/ontologies/ExtruOnt/components4ExtruOnt#hasSensor
› http://www.example.org/Pianism/OntoPianismERP.owl#erpTopDataProperty
› http://www.example.org/Pianism/OntoPianismERP.owl#erpExtruderHeadName

Figure 89 - OntoPianismERP Pitfalls P08

Minor Pitfalls P13 – Identifies a set of data properties as not having **"Inverse relationships not explicitly declared**" as shown in Figure 90. The created data properties do lack inverse relationships since they were not applicable for the present use case. As such, these minor pitfalls were ignored.



Results for P13: Inverse relationships not explicitly declared. **29 cases | Minor**

This pitfall appears when any relationship (except for those that are defined as symmetric properties using owl:SymmetricProperty) does not have an inverse relationship (owl:inverseOf) defined within the ontology.

• OOPS! has the following suggestions for the relationships without inverse:
› http://www.w3.org/2006/time#intervalIn could be inverse of http://www.w3.org/2006/time#intervalDisjoint
› http://www.w3.org/2006/time#intervalDisjoint could be inverse of http://www.w3.org/2006/time#intervalEquals

• Sorry, OOPS! has no suggestions for the following relationships without inverse:
› http://www.example.org/Pianism/OntoPianismERP.owl#erpRequestDate
› http://www.example.org/Pianism/OntoPianismERP.owl#erpOccurred
› http://www.example.org/Pianism/OntoPianismERP.owl#erpSpecifiesUseOf
› http://www.example.org/Pianism/OntoPianismERP.owl#erpTopObjectProperty
› http://www.example.org/Pianism/OntoPianismERP.owl#erpInterventionTime
› http://www.example.org/Pianism/OntoPianismERP.owl#erpResolutionTime
› http://www.semanticweb.org/katiaaleid(1181473)/ontologies/2019/10/ontology-extension#extTopObjectProperty
› http://www.example.org/Pianism/OntoPianismERP.owl#erpOperationFromOrder
› http://bdi.si.ehu.es/bdi/ontologies/ExtruOnt/components4ExtruOnt#hasSensor
› http://www.w3.org/2006/time#timeZone
› http://www.w3.org/2006/time#hasDurationDescription
› http://www.w3.org/2006/time#inTemporalPosition
› http://www.w3.org/2006/time#dayOfWeek
› http://www.w3.org/2006/time#hasTime

Figure 90 - OntoPianismERP Pitfalls P13

Minor Pitfalls P22 – Indicates that "Using different Naming Conventions in the Ontology" should be corrected, as shown in Figure 91. But the hints given are scarce and don´t provide enough guidance to perform any corrective action. The ***OntoPianismERP*** was properly built using a defined naming convention strategy in **10.4.2 Manual Evaluation**. This pitfall might be originated from the imported ontologies which have distinct naming conventions. This pitfall was therefore ignored.

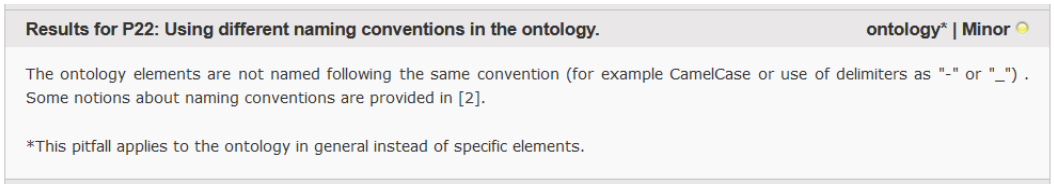


Figure 91 - OntoPianismERP Pitfalls P22

Suggestions – The 19 suggestions made by OOPS! shown in Figure 92 are all related to object properties of imported ontologies, and therefore also ignored.

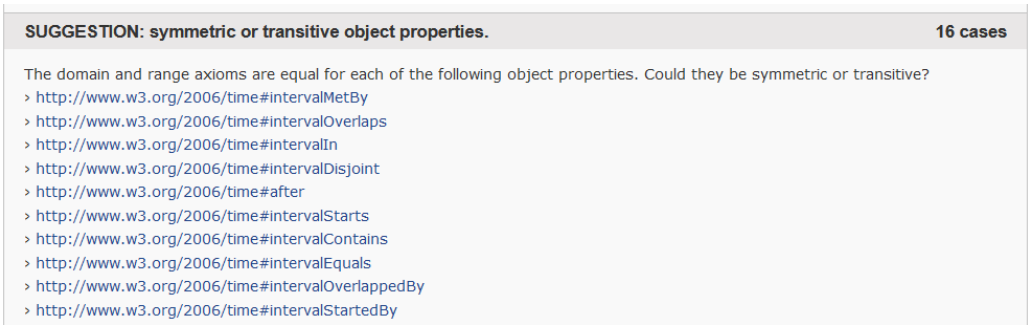


Figure 92 - OntoPianismERP Suggestions

10.4.4 Transformation and Analysis of Data Samples

The data collected from the Web Service described in 6.1.1 Requirements was fed to ***J2OIM*** (Nobre, Canito, Neves, Corchado, & Marreiros, 2022), specifically developed for this purpose, performing the transformation of real-world time-sensitive sensor data collected from industrial equipment and contextual information from existing management software. This tool managed to transform all raw data into the semantic representation defined in both ontologies, ***OntoProcessMapping*** and ***OntoPianismERP***, creating the OWL2 structured file ***OntoPianismIndividuals***. This later file contained all the concepts covered by the two former ontologies representing all the collected data into a semantic representation that accounts for its temporally dynamic origin.

The semantic data corresponding to sensor readings represented temporally based on the developed ontologies is shown in Figure 93 using the Protégé (Stanford Center for Biomedical Informatics Research, 2021) tool.

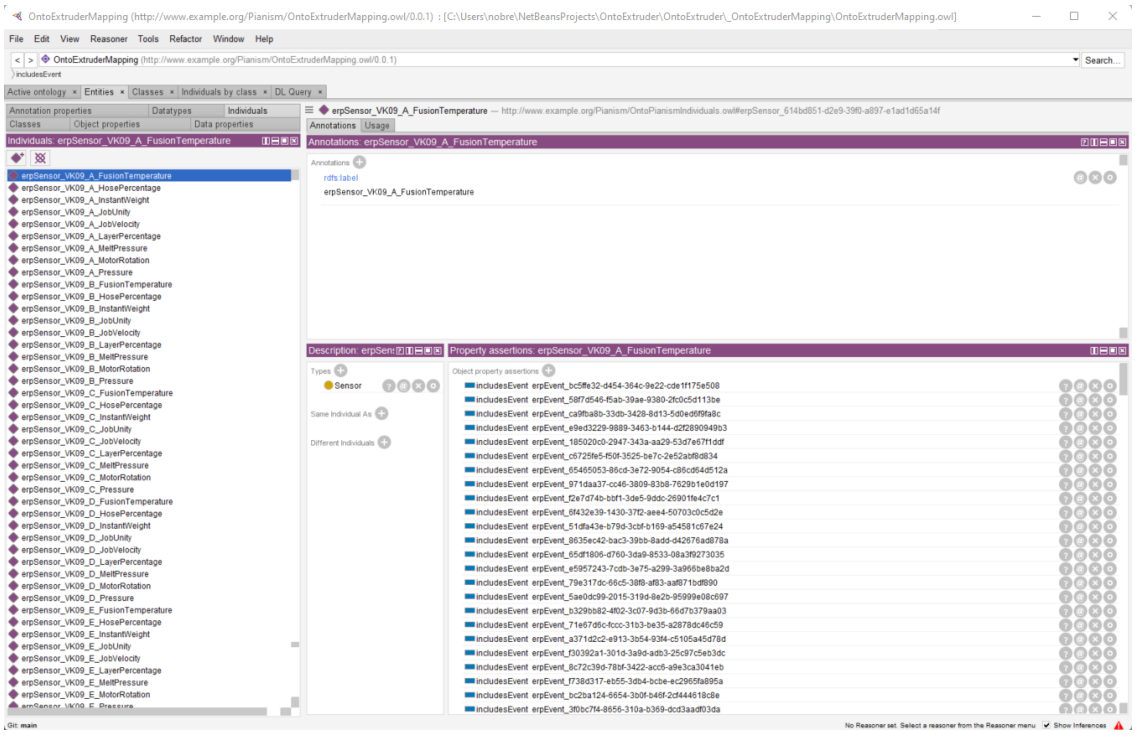


Figure 93 - Semantic transformed data in Protégé

J2OIM also enables the transformation of the collected data into a dedicated triple store, thus the semantic data is stored in the form of triples (*subject-predicate-object).* The usage of a triple store provides an alternative storage to the former OWL2 Ontology persistence of data. The triple stores search engine allows the usage of SPARQL's (W3C, 2021) query language to execute queries to retrieve data.

The semantic data corresponding to sensor readings represented temporally based on the developed ontologies is shown in Figure 94, representing the data in the Apache Jena Fuseki (Apache Jena, 2021) query interface.

QUERY RESULTS

Table | Raw Response

Showing 1 to 50 of 716 entries

Search: Show 50 entries

	subject	predicate	object
1	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_ffdadfcb-c4d1-3675-9142-5818a5d27fb1>
2	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_fe87baa9-0bd0-3600-9fd1-91d5a27d95c7>
3	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_fe6008d4-9e8b-323e-b524-429b2188306a>
4	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_fe018aab-903c-30ae-88af-6350dafe8afa>
5	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_fdc0a1d8-ac43-3dfe-b90e-4f065f6f2099>
6	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_fdae9037-5a11-376b-bbea-cea7a02b2211>
7	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_fda2560a-04cd-3de2-8644-fbb6715c9f33>
8	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpSensor_614bd851-d2e9-39f0-a897-e1ad1d65a14f>	<http://www.loa-cnr.it/ontologies/DUL.owl#includesEvent>	<http://www.example.org/Pianism/OntoPianismIndividuals.owl#erpEvent_fd63df03-b543-3fe6-be65-39412d2a9249>

Figure 94 - Semantic transformed data in Fuseki Triple Store

10.4.5 Scientific Contribution WorldCIST Conference Proceeding

In the year 2021, a conference paper was submitted to "Information Systems and Technologies: WorldCIST 2022"[9] , to be presented at “WorldCist'22 - 10th World Conference on Information Systems and Technologies”, held in Budva, Montenegro, 12 - 14 April 2022. The paper was titled “***Applying time-constraints using ontologies to sensor data for predictive maintenance***” (Nobre, Canito, Neves, Corchado, & Marreiros, 2022), and explores the current state of the application of ontologies in the domain of the representation of time-sensitive data and presents a proposal on how to apply a data model defined in an ontology to raw data.

The proposed data model transformed non-interconnected time-sensitive data from multiple sensors collected by PIANiSM (PIANiSM, 2020) and transposed the data into an file with a OWL2 structure representing the data in a structured hierarchy of triples (subject-predicate-object), representing time-sensitive data according to the 4d fluents pattern. This paper addresses the workings of ***J2OIM*** and its capacity to process raw data collected from a plastic extrusion

[9] Information Systems and Technologies: WorldCIST 2022 [Online] Available at http://worldcist.org/ (accessed 31-12-2022)

process and perform transformations to represent the data semantically and temporally. The capabilities of the ***J2OIM*** tool regarding the applicability of data transformation based on configuration files were comprehensively detailed to demonstrate the potential application in similar conditions.

The paper was accepted on December 31, 2021[10]. It should be noted, however, that the paper was submitted at a stage in which the proposed ontology was not yet fully developed, and the paper's main purpose was to serve as a proof of concept that it was feasible to transform static raw data based on an ontology representing the same data dynamically and temporally.

10.4.6 Scientific Contribution IOS Press Journal Article

In the year 2022, a second work, this time a journal article, was submitted to be published in the Integrated Computer-Aided Engineering Journal[11] (ICAE), part of the IOS Press[12] catalogue of journals. IOS Press is an international publishing house producing content covering science, technology, and medicine. This journal article was titled "***Using time-constraints in sensor data to detect ontology evolution for predictive maintenance"***, which further detailed the mechanisms and applications of ***J2OIM*** as part of a larger ecosystem of semantic tools. The semantically described data transformed by ***J2OIM*** was then used to guide an ontology evolution process in order to reflect the changes in ontology over time that may have been brought out by experiences with feature engineering, fault identification and variations in data sources. Feature engineering is the process of using domain knowledge to extract new attributes from raw data, to use these additional attributes to improve the quality of results from a machine learning process (Holbrook & Cook, 2022). Here, the ***J2OIM*** tool was configured to generate new features by combining or extending existing ones.

In order to assess changes in ontologies and verify when these changes have occurred, ***J2OIM*** worked alongside a second tool, the *TIme Constrained instance-guided Ontology evolution* (***TICO***) tool. This tool was capable of analysing a set of ontology individuals and determine in which ways they differ from the definitions present in the ontology, and modifies the ontology to reflect those changes. As such, this article showed a practical application of the ***J2OIM*** tool, as provider of different sets of ontologies with heterogenous samples of individuals by introducing variations in the ontologies' individuals by means of the generation of features according to the necessary of data to satisfy feature engineering.

---

[10] Applying Time-Constraints Using Ontologies to Sensor Data for Predictive Maintenance [Online] Available at https://link.springer.com/chapter/10.1007/978-3-031-04819-7_38 (Accessed: 17-05-2022)

[11] Integrated Computer-Aided Engineering [Online] Available at https://www.iospress.com/catalog/journals/integrated-computer-aided-engineering (accessed 16-09-2023)

[12] IOS Press [Online] Available at https://www.iospress.com/ (accessed 16-09-2023)

10.5 Summary

This chapter demonstrated the various actions taken to validate the domain ontology ***OntoProcessMapping*** and its auxiliary ontology ***OntoPianismErp***. The developed OWL files were verified following the D101 guidelines. Additionally, the online tool OOPS! was used to evaluate existing pitfalls, which were analysed and fixed whenever possible.

Data samples collected from Sistrade ERP, describing real-time plastic extrusion equipment manufacturing processes, were transformed using the JSON to Ontology Instance Mapper (***J2OIM***) tool, and the resulting ontology ***OntoPianismIndividuals*** representing the semantic data was analysed in the Protégé tool.

Finally, the application of the ontologies and the tools developed to support them resulted in the publication and submission of scientific papers.

11 Conclusion and future work

This dissertation focused as a primary study on the design and development of a domain ontology named ***OntoProcessMapping*** for the field of Predictive Maintenance, which shall be used to describe real-world time-sensitive data collected from industrial machinery sensors. This domain ontology was designed encompassing and reusing several publicly available ontologies, which already contained definitions regarding manufacturing processes, time constraints, and plastic extrusion processes. To cover the scope of the PIANiSM project (PIANiSM, 2020) a new ontology, named ***OntoPianismERP,*** was designed and developed to represent the existing concepts already available through the project's Web Services.

The resulting ***OntoProcessMapping*** domain ontology acts as a bridge by grouping and connecting the public ontologies and the developed ***OntoPianismERP*** ontology into a single ontology that represents a broad range of concepts necessary to describe the plastic extrusion process and thus enabling the representation of this manufacturing process semantically.

The domain ontology introduced new concepts to allow the temporal representation of time-sensitive data following the 4d fluents pattern to represent the persistence of objects through time by reifying only the temporal part of the object. This feature not only allowed the decoupling of occurrences and values from the originating sources but was the fundamental source for obtaining new knowledge, by representing once static data in a temporal dynamic fashion, in such a way that it can be used as input for the predictive ML and DM algorithms.

The ***OntoProcessMapping*** domain ontology was designed to model the manufacturing process of plastic extrusion machines, and to satisfy the requirements of PIANiSM project (PIANiSM, 2020). However, the foundations to represent time-sensitive data introduced in this ontology may be re-used in different manufacturing scenarios. Thus, with a reduced effort the ***OntoProcessMapping*** domain ontology could be expanded to cover other manufacturing domains where data is acquired from sensors and must be related to events occurring during the manufacturing process. Eventually, the developed ontology could be multipurpose and span across multiple domains.

This former assumption is supported through the development of the ***JSON to Ontology Instance Mapper tool (J2OIM)***, which established an architecture for the transformation of sensor data acquired directly from industrial machines and contextual information facilitated by ERP software into semantic entities with a temporal component that can be reasoned with. ***J2OIM*** allows parameterizable transformation of JSON instances into different ontological instances, which can be either temporally dynamic or static. The generated instances populate two types of repositories, an Ontology File and Triple-Store, this new knowledge can ultimately be used alongside ML and DM activities to bring forth predictive maintenance practices.

A solid conclusion taken from this development is the creation of a very high number of instances required for temporal representation, events, and intervals.

Even considering a reduced set of sample data, approximately a million new instances were created to temporally represent sensor readings, occurrences, and manufacturing operations.

Undoubtedly, the sharp increase in number of instances shall also increase processing and reasoning times and will require more complex queries to group and extract data, but the increase in individuals is a direct consequence of having chosen 4d fluents and that was a thoughtful decision and an expected consequence, especially considering that there were alternatives that would multiply the number of instances even more.

Nevertheless, the increase in instances contributed to valuable knowledge not priorly present in the raw data and is most likely to impact the future application of ML and DM algorithms targeting the structured semantic data to evaluate and predict the occurrence of equipment malfunction in the desired scope of PdM.

The execution of future tests should be able to determine the impact of the present approach, the usage of the structured semantic data in PdM, and should also serve to further assess the impact of the domain ontology's application.

As a next step, applying supervised and unsupervised ML algorithms to the temporal data, would provide means to identify and classify patterns of abnormal values or values with a deviation compared with data obtained in regular manufacturing cycles and standard data supplied by the manufacturer's reference values. These abnormalities shall identify the potential future incidents in component malfunctions.

The abnormal behaviours can be matched with the data available from the ERP and thus also be directly related to the manufacturing order being processed and the occurrences observed in the same temporal interval. The path for the effective evaluation of the current working conditions of machinery components and the prediction of its time to a potential failure is thus facilitated.

The present work culminated in delivering structured semantic data, focussing on the temporal representation of events, in an Ontology File and a Fuseki Triple Store, being both formats suitable to be utilized for ML and DM algorithms.

As future work – beyond the aim of the current one – would be the study of which are most adequate ML and DM algorithms to be applied to the data to obtain the results leading to the prediction of the component faults, and ultimately the PdM itself.

In conclusion, as extensive as the presented work revealed itself, in terms of achievements, as previously detailed, the future use in terms of the application of ML and DM algorithms for PdM is enormous and supported by a solid base of semantic data.

12 References

Alamri, A., & Bertok, P. (2012). Distributed Store for Ontology Data Management. In *Computer and Information Science* (pp. 15–35). Berlin : Springer-Verlag.

Aleid, K. (2020). Development of an Integrated Ontology to Enhance Interoperability for Airports' Security Solutions. ISEP.

Aleid, K., & Canito, A. (2020). *Airport Security Interoperability Integrated Ontology*. Retrieved 12 15, 2021, from https://www.gecad.isep.ipp.pt/ASIIO_SITE/index-en.html

Alias-i, Inc. (2019). *API Tutorials*. Retrieved 02 24, 2021, from http://www.alias-i.com/lingpipe/demos/tutorial/read-me.html

Anagnostopoulos, E., Batsakis, S., & Petrakis, E. (2013). CHRONOS: A reasoning engine for qualitative temporal information in OWL. *Procedia Computer Science*. doi:https://doi.org/10.1016/j.procs.2013.09.082

Apache Jena. (2021, 11 03). *Apache Jena Fuseki*. Retrieved from Apache Jena: https://jena.apache.org/documentation/fuseki2/index.html

Auer, S., Lehmann, J., Ngomo, N. A.-C., & Zaveri, A. (2011). Introduction to Linked Data and its Lifecycle on the Web. Galway: Reasoning Web. Semantic Technologies for the Web of Data - 7th International Summer School 2011.

BSI, B. S. (2010). *Maintenance — Maintenance terminology.* London: British Standards Institution.

Burek, P., Scherf, N., & Herre, H. (2019). Ontology patterns for the representation of in time. *JOURNAL OF BIOMEDICAL SEMANTICS, 1*(10).

Cambridge Semantics. (2021). *Introduction to the Semantic Web*. Retrieved 02 01, 2021, from https://www.cambridgesemantics.com/blog/semantic-university/intro-semantic-web/many-names-semantic-web/

Canito, A., Corchado, J., & Marreiros, G. (2021). A systematic review on time-constrained ontology evolution in predictive maintenance. *Artificial Intelligence Review*. doi:https://doi.org/10.1007/s10462-021-10079-z

Canito, A., Corchado, J., & Marreiros, G. (2021). Bridging the gap between domain ontologies for predictive maintenance with machine learning. *Advances in Intelligent Systems and Computing, 1366*.

Canito, A., Corchado, J., & Marreiros, G. (2021). Bridging the gap between domain ontologies for predictive maintenance with machine learning. *Advances in Intelligent Systems and Computing*, 533-543.

Cheong, H. (2019). Translating JSON Schema logics into OWL axioms for unified data validation on a digital data platform. *ScienceDirect*, 183-188.

Cho, S., May, G., & Kiritsis, D. (2019). A Semantic-driven Approach for Industry 4.0. *15th International Conference on Distributed Computing in Sensor Systems (DCOSS),*, 47–354.

Creative Commons. (2022). *crative commons*. Retrieved 02 07, 2022, from https://creativecommons.org/licenses/by-sa/4.0/

Danyliw, J. R. (2007). The incident object description exchange format,IETF RFC 5070. *doi: 2070-1721, 62*(1), 27–40.

Deshmukh and Garg. (2006). Maintenance management: literature review and directions. *Journal of Quality in Maintenance Engineering, 12*(3), 205–238.

Dhillon, B. S. (2006). Maintainability, maintenance, and reliability for engineers. CRC Press.

Falconer, S. (2010). *OntoGraf*. Retrieved 05 14, 2022, from https://protegewiki.stanford.edu/wiki/OntoGraf

GHORBEL, F., HAMDI, F., & METAIS, E. (2019). Ontology-Based Representation and Reasoning about Precise and Imprecise Time Intervals. *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 1–8.

Giustozzi, F., Saunier, J., & Zanni-Merk, C. (2018). Context Modeling for Industry 4.0: an Ontology-Based Proposal. *Procedia Computer Science, 126*, 675-684.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies, 43*(5-6), 907-928.

Holbrook, R., & Cook, A. (2022, 06 01). *Creating Features*. Retrieved from https://www.kaggle.com/code/ryanholbrook/creating-features/tutorial

Hwang, C.-L., Lai, Y.-J., & Liu, T.-Y. (1993). A new approach for multiple objective decision making. *Computers & Operations Research, 20*(8), 889-899.

Koen, P., Ajamian, G., Burkart, R., Clamen, A., Davidson, J., D'Amore, R., . . . Wagner, K. (2001). PROVIDING CLARITY AND A COMMON LANGUAGE TO THE "FUZZY FRONT END". *RESEARCH • TECHNOLOGY MANAGEMENT, March-April*, 46-55.

Kootanaee, J. A., Babu, K. N., & Talari, F. H. (2013). Just-in-Time Manufacturing System: From Introduction to Implement. *International Journal of Economics, Business and Finance, 1*(2), 07 - 25.

Mazzola, L., Kapahnke, P., Vujic, M., & Klusch, M. (2016). *CDM-Core: A Manufacturing Domain Ontology in OWL2 for Production and Maintenance.* Saarbruecken, Germany: Center for Artificial Intelligence (DFKI).

Michael, S. (2007). *OntoViz*. Retrieved 05 15, 2022, from https://protegewiki.stanford.edu/wiki/OntoViz

Nobre, A., Canito, A., Neves, J., Corchado, J., & Marreiros, G. (2022). Applying Time-Constraints Using Ontologies to Sensor Data for Predictive Maintenance. *WorldCIST 2022: Information Systems and Technologies*, 390-400.

Noy, F. N., & McGuinness, L. D. (2001). Ontology Development 101: A Guide to Creating Your First Ontology. *Stanford Knowledge Systems Laboratory*.

OGC & W3C. (2020). *Time Ontology in OWL*. Retrieved 12 15, 2021, from https://www.w3.org/TR/owl-time/

Oracle. (2021). *https://www.oracle.com/java/index.html.* Retrieved 02 23, 2021, from https://www.oracle.com/java/

Osterwalder, A. (2021). *Hi! I'm Alex Osterwalder*. Retrieved 02 15, 2021, from http://alexosterwalder.com/

Panov, P., & Stefan, J. (2020). *The OntoDM Ontology*. Retrieved 11 01, 2020, from http://www.ontodm.com/doku.php

Parmenides. (5th c. BCE). *On nature.*

Pavić, Z., & Novoselac, V. (2013). Notes on TOPSIS Method. International Journal of Research in Engineering and Science (IJRES), 1(2), 05-12.

Pijl, P. v. (2016). *How to really understand your customer with the Value Proposition Canvas*. Retrieved 02 15, 2021, from https://designabetterbusiness.com/2017/10/12/how-to-really-understand-your-customer-with-the-value-proposition-canvas/

Poveda, M. (2021). *OOPS! (OntOlogy Pitfall Scanner!)* . Retrieved 02 07, 2022, from http://oops.linkeddata.es/response.jsp#

Preventis, A., Marki, P., Petrakis, E. G., & Batsakis, S. (2011). CHRONOS: A tool for handling temporal ontologies in protégé. *International Journal of Artificial Intelligence Tools*, 212.

Preventis, A., Marki, P., Petrakis, E. G., & Batsakis, S. (2012). CHRONOS: A tool for handling temporal ontologies in protégé. *International Journal of Artificial Intelligence Tools*.

R.Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition, 5*(2), 199-220.

Raad, J., & Cruz, C. (2015). A Survey on Ontology Evaluation Methods.

Ramírez-Durán, V., Berges, I., & Illarramendi, A. (2019). *ExtruOnt: An RDF/OWL schema for representing an extruder*. Retrieved 12 15, 2021, from http://siul02.si.ehu.es/bdi/ontologies/ExtruOnt/docs/

Runeson, P., & Höst, M. (2019). Guidelines for conducting and reporting case study. *Empir Software Eng, 14*(2), 131–164.

Sbai, S., Louhdi, R. M., Behja, H., & Chakhmoune, R. (2019). JsonToOnto: Building Owl2 Ontologies from Json Documents. *International Journal of Advanced Computer Science and Applications, 10*(10), 213-218.

Schreiber, G., Akkermans, H., Anjewierden, A., Hoog, R., Shadbolt, N., & Wielinga, B. (2000). Knowledge Engineering and Management: The Common KADS Methodology. *99*.

Sharma, R. (2019). *Top 15 Sensor Types Being Used Most By IoT Application Development Companies*. Retrieved February 01, 2021, from https://www.finoit.com/blog/top-15-sensor-types-used-iot/

Sistrade, ISEP, GECAD, Experis IT. (2020). Predictive and Prescriptive Automation Smart Manufacturing, State of the Art.

Stanford Center for Biomedical Informatics Research. (2020). *Protégé*. Retrieved 02 22, 221, from https://protege.stanford.edu/

Stanford Center for Biomedical Informatics Research. (2021). *Protégé*. Retrieved 02 22, 2021, from A free, open-source ontology editor and framework for building intelligent systems: https://protege.stanford.edu/

Suárez-Figueroa, M. C., Gómez-Pérez, A., & Villazón-Terrazas, B. (2009). How to Write and Use the Ontology Requirements Specification Document.

Syed, Z., Pädia, T., Mathews, L., & Joshi, A. (2016). UCO: A Unified Cybersecurity Ontology. *AAAI Work. - Tech. Rep, WS-16-01*, 195–202.

The Apache Software Foundation. (2020). *Apache NetBeans*. Retrieved 02 23, 2021, from https://netbeans.apache.org/

The Apache Software Foundation. (2021). *Apache Jena Fuseki*. Retrieved 11 03, 2021, from https://jena.apache.org/documentation/fuseki2/

The Apache Software Foundation. (2021). *Apache Maven Project*. Retrieved 02 23, 2020, from https://maven.apache.org/#

The Apache Software Foundation. (2021). *Jena Ontology API*. Retrieved 02 23, 2021, from https://jena.apache.org/documentation/ontology/

w3.org. (2004). *Resource Description Framework (RDF)*. Retrieved 04 25, 2022, from https://www.w3.org/RDF/

w3.org. (2009). *Web Ontology Language (OWL)*. Retrieved 14 25, 2022, from https://www.w3.org/OWL/

w3.org. (2012). *OWL 2 Web Ontology Language*. Retrieved 04 25, 2022, from https://www.w3.org/TR/owl2-primer/#Object_Properties

W3C. (2011). *Semantic Sensor Network Ontology*. Retrieved 12 15, 2021, from https://www.w3.org/2005/Incubator/ssn/ssnx/ssn

W3C. (2013). *SPARQL 1.1 Protocol*. Retrieved 11 03, 2021, from https://www.w3.org/TR/sparql11-protocol/

W3C. (2021, 11 03). *SPARQL 1.1 Protocol*. Retrieved from SPARQL 1.1 Protocol: https://www.w3.org/TR/sparql11-protocol/

Warwick Manufacturing Group. (2007). PRODUCT EXCELLENCE USING SIX SIGMA. In U. o. Warwick, *Quality Function Deployment.* Coventry: School of Engineering University of Warwick.