Politecnico di Torino

Computation of Semantic Similarity Between Products by Analysing Data Collected by PLM and MES Systems

Supervisors
Prof. Franco Lombardi
Prof. Giulia Bruno

Candidate
Hassan Khattab

Academic Year 2020-2021

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to Politecnico di Torino for giving me the chance to enjoy such a unique academic experience. I would like also to thank all my professors and especially Prof. Franco Lombardi and Prof. Giulia Bruno who guided me all the way to reach my graduation.

I am extremely grateful tor my parents for their indispensable and continuous support despite the long distance that separated us during my university studies.

I want to thank my friends in Italy, Lebanon and Germany who gave me additional support during my journey and made it so special.

Lastly, I would like to thank EDISU PIEMONTE for the financial support offered to me during my postgraduate studies

ABSTRACT

The extensive adoption of digital technologies by manufacturing companies, imposed by the global competition and the advent of industry 4.0, has led to the generation of huge amounts of data. If well exploited to create useful knowledge, such data can provide the companies with superior and sustainable competitive advantage. For this reason, there is an increasing interest in manufacturing companies to develop advanced knowledge management systems. Among different companies, those operating in one-of-a-kind production find it crucial to effectively manage their knowledge because they are under the pressure of delivering high quality products in a short time and at low cost.

To this aim, several researches have been conducted to effectively manage the knowledge and most of them highlighted the importance of integrating information coming from different sources.

This thesis addresses the problem of identifying similar products by proposing a knowledge management framework that exploits the data integration between Product Lifecycle Management (PLM) and Manufacturing Execution System (MES). It focuses on the application of semantic measurements that use ontologies to calculate product similarities.

The effectiveness of semantic similarity method through the adoption of information content approach is demonstrated by applying it on real data belonging to a company that produces prototypes components in the automotive sector and its products are highly customized and should match with individual customer needs. The aim is to capture the implicit knowledge embedded inside employees minds and make it transferable and reusable by other employees to enhance the way of defining production cycles for new products and to make them more accurate.

Keywords: Industry 4.0, One-of-a-Kind-Production, Knowledge Management, PLM, MES, Semantic similarity

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

INTRODUCTION

Globalization and the increased competition in manufacturing industry is pushing toward accelerating product development and optimizing manufacturing processes to obtain products with high quality in a short period of time. To tackle with these challenges, companies are adopting the adequate technological tools and manufacturing approaches to maintain their position in the market by minimizing waste and continuously improving product quality. The application of these approaches and tools is more beneficial to industries with high production volumes and low to medium product variability. However, we are living in an era where customers require products that are tailored to their needs and the One-of-a-Kind production (OKP) paradigm is going to dominate the industry in the future according to the experts.

In order for OKP companies to satisfy customer needs without sacrificing high quality and competitive pricing, they rely on the implementation of Information Technology (IT) tools such as Enterprise Resource Planning ERP, Manufacturing Execution Systems (MES), and Product Lifecycle Management (PLM) as they are useful to cope with the competition and to work effectively during the design, production and industrial processes (Ben Khedher et al. 2011). PLM main objective is to manage product related information throughout the whole product lifecycle. ERP provides an enterprise database that allows to trace all the activities related to sales, finance, marketing, human resource, finance run by the company (Umble et al. 2003). MES aims at evaluating the optimal planning sequence considering process features as well as allocation of resources and materials considering the capacity of workstations, set up and processing time. It also manages the data flow that regards process performance and product quality from the shopfloor (D'Antoni et al., 2015). Although, ERP, PLM and MES provide useful tools by themselves, their efficacy can be increased if they are able to exchange information with each other. The integration between PLM and MES becomes crucial in OKP environment due to the concurrent execution of design and production activities which requires the maximum level of coordination due to the frequent modifications because of the continuous customer intervention and the uncertainties that may arise during manufacturing. The lack of such integration results in significant losses and delays and the appropriate integration creates competitive advantage for the company.

In addition to the integration, the storage of data generated by these systems and transforming them into useful knowledge for future use can accelerate drastically product development because it allows the company to leverage on the acquired knowledge in an effective way. The key step in knowledge reuse is the identification of similar products because they can use similar design and manufacturing processes. However, there is no unique approach for estimating this similarity and there are different methods whose effectiveness rely on the context and the field of application. Shape-based similarity are extensively used by many companies and it works by comparing 3D models of products in order to assign similarity values. This approach has a big limitation because the shape contains only a small part of product related knowledge. As a result, a more exhaustive similarity tool is required in order to consider as much knowledge as possible in similarity assessment.

By this thesis, we aim to support OKP companies to increase their responsiveness in such a volatile environment by proposing a semantic similarity framework that is applied on domain ontologies of a Knowledge Base System (KBS) that integrates data from MES and PLM systems. This enables the reuse of existing manufacturing and design knowledge during the development of new product by identifying similar products manufactured in the past. This saves reasonable amount of time as it avoids the duplication of design efforts for similar products and because the designer is made aware at an early stage of manufacturing issues related to similar products. It is also useful for reducing waste by preventing the replication of errors either at the design or at manufacturing stage. Moreover, it makes the company able to retain its knowledge and transfer it within the organization and decreases the information overload for its experts as they will be only focusing in solving new problems.

The thesis starts by introducing industry 4.0 and the information systems implemented by companies to cope with industry trends in addition to the description of the characteristics of one-of-a-kind production (OKP). The second part provides an overview about knowledge management and highlights its key role inside the organization and it briefly describes the different approaches in knowledge reuse for manufacturing companies. The third part describes the different semantic similarity measures covered by the literature as well as their accuracy. The fourth part explains the methodology of identifying similar products by using data available in a knowledge base system (KBS) that integrates MES and PLM. The fifth part shows how the proposed similarity methodology is applied on data belonging to a typical OKP company that operates in Italy and produces car prototypes for famous car manufacturers. Lastly, we highlight the usefulness of and the limitations of the proposed approach and how it can be furtherly improved.

Chapter 1: BACKGROUND

1.1 Industry 4.0

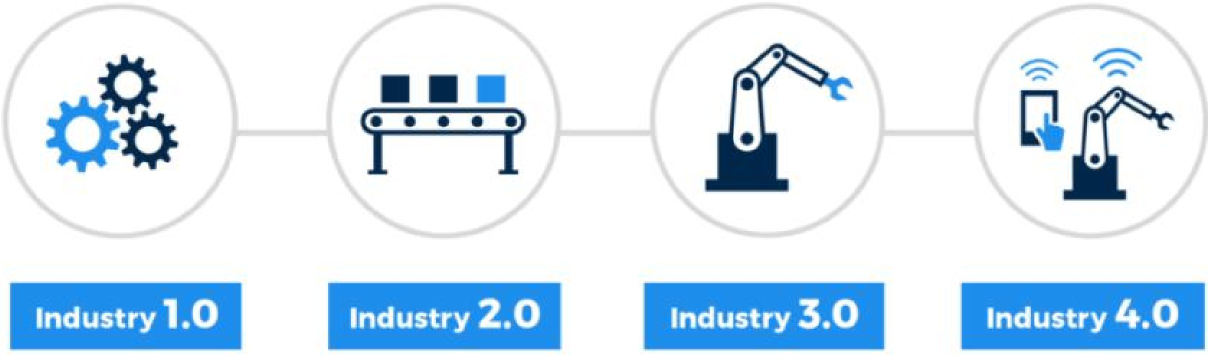


Figure 1 The four industrial revolutions

Through the history, the manufacturing industry has reinvented itself many times in order to respond to the needs of the society and to fully exploit the available technologies. The first industrial revolution dates to the late 18th century where steam engines and hydraulic power became widely used by manufacturers in their factories to make production more efficient. In the late 19th century, the second revolution started, and it is when electricity and assembly lines made mass production possible. The third revolution took place in the 60s when advancement in computing enabled the programming of machines and networks. Today, we are living in the fourth industrial revolution which is changing radically the way companies operate by focusing on connecting digital and physical worlds. The following are the main aspects that characterizes industry 4.0:

- Interconnectivity: The ability of the assets and resources of the production chain to exchange information with internal and external systems through the use of a data exchange network. This concept does not concern only the factory of the manufacturing company, but it also includes all the other actors along the value chain such as suppliers and customers.
- Decentralization: The components of the cyper-physical system can recognize any anomalies in the processes and modify their behaviour in autonomous ways.
- Remote interaction: The possibility to monitor, through remote access, the processes in order to collect data that allow, in the event of malfunctions, to intervene proactively.
- Real-time processing: The presence of functions that make it possible to collect information in real time so that it is possible to respond by taking immediate actions.
- Modularity: The possibility of modifying production mechanisms on the basis of variations in demand by exploiting an increasingly integrated value chain from the point of view of information sharing.
- Sustainability: An element that is not purely technological but is no less important for this. It refers to environmental and social elements such as the optimization of energy and resource consumption and the improvement of working conditions.

- Interoperability: Ability of two or more systems belonging to different companies to exchange data in order to create networks of companies that can also extend beyond the borders of the national territory in order to allow even medium-small companies to increase their competitiveness.

The fourth industrial revolution is commonly associated with a set of technologies that are the following: Internet of Things (IoT), Cloud Computing, Additive Manufacturing, Big Data Analytics, Collaborative Robots, Augmented Reality and Cybersecurity. They are described as follows:

- Internet of Things (IOT): The Internet of Things is configured as a network of physical objects that incorporate specific technologies for the detection and transmission of data through an Internet network. Through the application of appropriate sensors, it is possible to integrate the virtual world of IT with the real world, thus shaping a real ecosystem in which a product, for example, becomes capable of transmitting useful information anywhere and at any time about its state or the surrounding environment in real time. There are various benefits that come from the use of the IoT in the production sectors;. For instance, by installing a set of sensors in the various phases that make up the production process, it is possible to obtain data relating to any type of parameter, which allow, for example, to intervene promptly in cases of malfunction, avoiding incurring longer setup times and higher costs.
- Big Data Analytics: Data is one of the main pillars of the fourth industrial revolution as the digitization of the company leads to the creation of a large amount of data, which needs to be collected and analysed for supporting the company in important activities such as decision making or performance evaluation. One key advantage of this technology is allowing the prediction of malfunctions ex-ante as well as to monitoring, controlling measure the performance of production processes in real time. Furthermore, the Data Analytics can be beneficial in interacting with customers since it allows the in-depth study of them in order to understand their behaviours, trends and habits, thus favouring mass customization if it is one of the organizations objectives. In a nutshell, the correct analysis is the indispensable prerequisite for successfully implementing the digital transformation of the factory.
- Cloud Computing: Through this technology it is therefore possible to govern the enormous amount of data generated by the sensors characterizing the IoT described above. Cloud Computing tends to be configured as a service provided by third parties according to the methods, times and costs decided by the users themselves, absolving those who use it from any responsibility regarding management and maintenance, since it is requested as the only an internet connection is required. It is flexible technology that allows to modify the contractual conditions agreed with the supplier in real time on the basis of the needs. The main criticality deriving from the adoption of a Cloud infrastructure concerns, without any doubt, data security: it will in fact be the user's concern to verify the reliability of the provider of the service in order to guarantee correct treatment of the information transferred to the platform. virtual.
- Cybersecurity**:** The information generated by technologies such as the Internet of Things can be sensitive and it is used not only within the company, but also by other players, such as suppliers and end customers, which is why the guarantee of correct use and processing of data becomes essential. IT security does not

only concern the adoption of antivirus systems and protection from hacker attacks, but also includes activities such as the identification of what can be considered critical information, to limit access to authorized persons only and to take measures to prevent any cybercriminals from entering it.

- Additive manufacturing: Three-dimensional printers are not a recent generation innovation. However, their evolution over time has led to the possibility of producing any type of product with a consistent reduction of waste. The opportunity to shape a physical object starting directly from a digital representation represents a valuable opportunity for the pursuit of objectives such as time-to-market reduction, mass customization, reduction of production costs and stocks
- Collaborative Robots: These robots are designed with the aim of operating in close contact with humans by sharing workplaces with them. They are equipped with sensors that allow them to recognize the presence of any operators and to stop if there is a collision with one of them and to restart once the risk goes away. Consequently, the safety of workers is significantly affected, since these latest generation machines have special camera mechanisms that ensure continuous monitoring of the environment around them, drastically reducing the likelihood of serious accidents. At the same time, however, it must be emphasized that it is not possible to avoid any errors committed by humans, which is why it is necessary to train and train people to interface in the most suitable way with these new technologies. The main benefit of these "intelligent machines" consists in their ease of reprogramming which allows them to be placed in numerous work areas, giving flexibility to production processes.
- Augmented reality and wearable devices**:** Using these devices, it is possible to view the real world enriched with real virtual objects that allow the operator to get hold of a much greater amount of data than those he would have access without using these devices. This allows to significantly simplify very complex operations such as, maintenance and repairs; in fact, the ability to view detailed intervention methods while a critical activity is being carried out represents a huge opportunity made possible by the following technology. Some of these devices are "smart" glasses, bracelets and watches that enable analysing data and subsequently sharing them by the operator who wears them. In addition to the operational advantages, these devices can help measuring and monitoring the health of the worker through key indicators, such as the heartbeat, in order to minimize injuries and accidents at work.

1.2 Product Lifecycle Management (PLM)

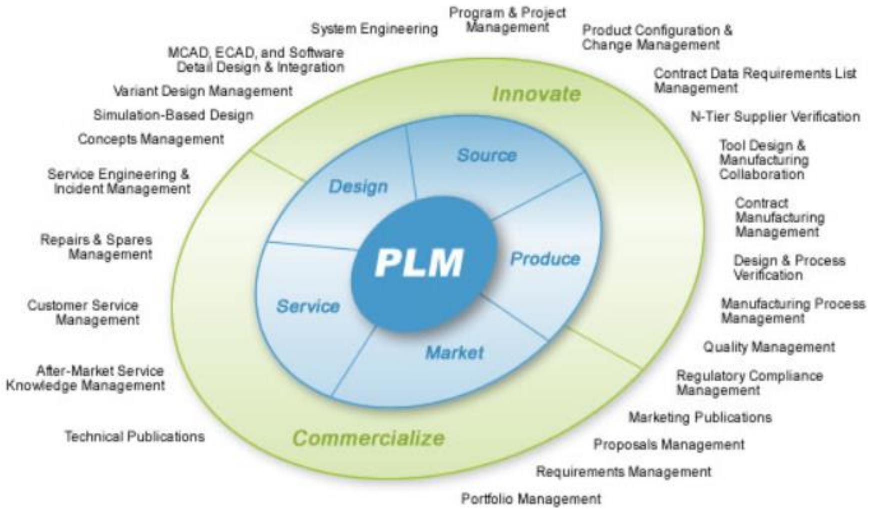


Figure 2 Stages of PLM

Product lifecycle management (PLM) is an approach to manage data and information related to the product and the resources involved in all stages of its life from dawn to dusk (Stark J., 2011). It connects data, processes, resources, and people together. To better understand it PLM we can break it into 3 pieces:

P: This letter represents the product, and it is the core of PLM as the product is the most important factor because it is the source of revenue for the company in one side and it satisfies customers' needs on the other side. Thus, it is a crucial part for the survival of the company because it is the main way by which company get financed to maintain its costs and also to grow in size. Product is the output of the company and indicates its capabilities in creating value abilities in responding to customer needs. That is why a consumer evaluate a certain company based on the quality, price and sometimes to the related services. Leading companies who can provide best products from customer perspective can enjoy higher market share than others with inferior products. Products can be tangible like an airplane or a car or intangible like a software or a service and sometimes can be a combination of both.

L: It refers to the lifecycle of the product who like human pass through different phases that can be grouped in three parts (Terzi et al, 2010): Beginning of life (BOL), which that consists of design and manufacturing activities after which the product is ready for consumer use ,Middle of life(MOL) that consists of usage, service and maintenance of the product and lastly End of life (EOL), that consists of the final stage of product at which the product is no longer useful and can be remanufactured, recycled reused or disposed. Companies are concerned about all these stages because at each of them the company can intervene to maximize value or cut costs

M: It is about the activities carried out by the companies to manage the product at each phase of its life to maximize their revenues, reduce the overall costs related to the product and improve the value of products for all the stakeholders involved. If the company is aware of all information related to the product it can intervene and take appropriate actions. For example, a company may have a product with superior quality but is less successful compared to other companies offering inferior products due to difficulty in using it or because of costly maintenance of the former and the ease of use and low maintenance of the latter. At this point the company can review the design to solve usability and maintenance problems and can improve its market position. Another example can be by a company focusing on the product quality and use durable and high performing materials but overlooking the sustainability of these materials which becomes a big issue at the end of product life. As a result, this product despite having good quality may be banned from being used in some countries with strict rules about protecting the environment.

1.3 Manufacturing Execution System (MES)

Figure 3 MES connecting production plant with ERP

Manufacturing execution system (MES) is the system used to monitor, control and manage data related to production processes from the time of receiving order until the product is finished. The main advantage of this system is to provide interface between business layer of the company and the shopfloor so that the management is made aware of all production status in real time and can act accordingly by taking the required decisions. Therefore, MES is usually integrated with ERP in order to be exploited in a good way because the raw data themselves are not sufficient if no corresponding actions are taken. To ensure such integration all MES systems should comply with ANSI/ISA-95 standard that was developed by International Society of Automation (ISA) in order to provide interface between control systems and enterprise system.

In addition to data collection that is the main function of MES, there the following:

- Planning and scheduling of the production
- Management of the availability of resources
- Labour management
- Quality management
- Process management

- Management and control of documents
- Management of maintenance activities
- Traceability of both finished and semi-finished products (WIP)
- Performance analysis

1.3.1 Integration between PLM and MES

A further exploitation of MES can be done by integrating it with PLM system because like ERP, shopfloor data can provide useful information to during product development phase. There is a growing interest about this topic and several research papers covered this integration, proposed frameworks for its application that and highlighted the benefits of such integration. (Taher et al., 2020) proposal is one of the most recent ones covering this topic and it addresses the application of the integrated system for OKP companies. In their work, they wanted to resolve the inefficient communication between the production and design department by providing a visual interface of product model between the two departments through a web-based CAD system. They used Aras innovator to implement PLM and Qcadoo for MES and the interface was obtained using WebTransCAD. As shown in the figure, the system works as follows: after the 3D model is created by the designers and stored in PLM database, it can be visualised immediately by manufacturing operators. In case of design errors, a 3D note function in X3D format is sent back to PLM to communicate the feedback to the designers. Lastly, the design is reviewed, modified and sent again to MES.

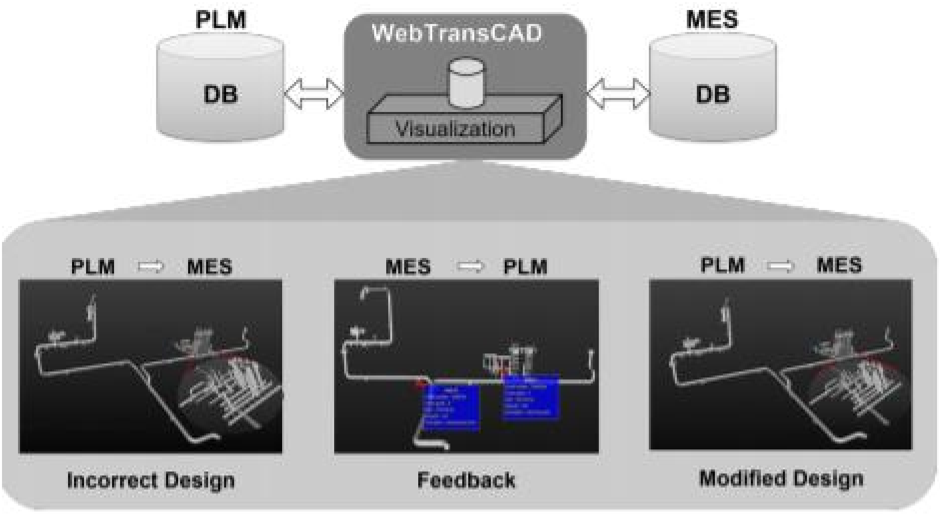


Figure 4 Integrating PLM and MES through The Visual interface framework (Taher et al., 2020)

(Lombardi et al., 2019) proposed the integration of PLM and MES as a tool for formalizing product knowledge also in the field of OKP for a company producing car prototypes to facilitate its reuse as well as transfer and to accelerate product development. Their framework is more extensive than the previous one which was only focused on the geometry of the product. As illustrated in the figure below, the central database integrates all product related data created and stored at PLM during design stage and manufacturing data generated during production phase. The exchange of data in real time between the two systems enhances feedback communication efficiency as designers can notice immediately any anomaly occurring at the shopfloor. Moreover, the stored data in the central database, allows the retrieval of previous products data that can be used for developing new products if they are similar to the old ones.

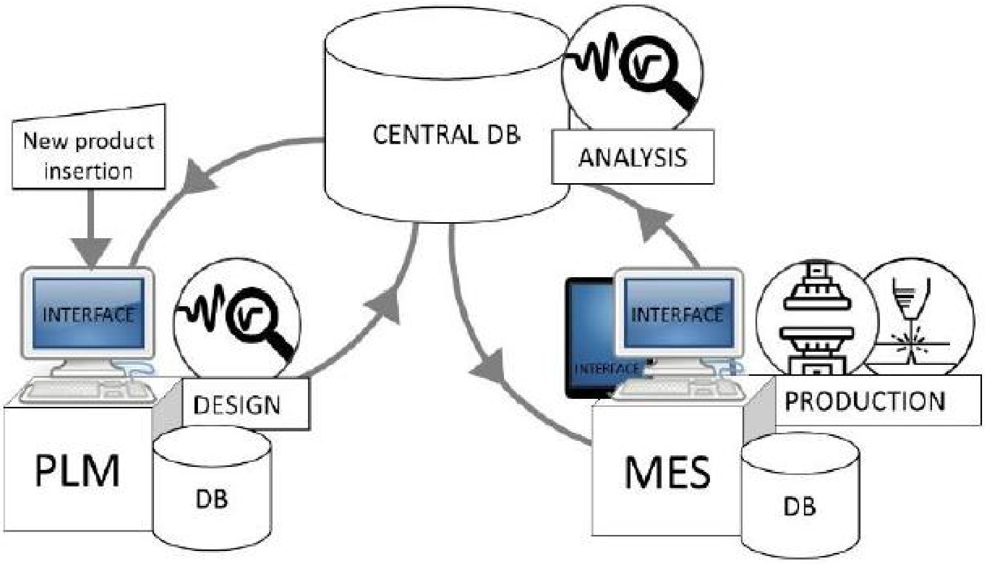


Figure 5 Integrating MES and PLM through a central database, (Lombardi et al., 2019)

1.4 Adoption of PLM and MES in Piedmont Region

To have a general overview about the adoption PLM and MES systems and the awareness of their benefits in Italy and in the Piedmont Region in specific, we can refer to the survey done by (D'Antoni et al., 2017) on 33 Italian companies operating in different sectors on which they were asked to indicate the information systems they are employing as well as their interest in the adoption of such systems. Moreover, companies were asked to indicate the expected benefit for the adoption of these systems both separately and integrated way.

PLM:

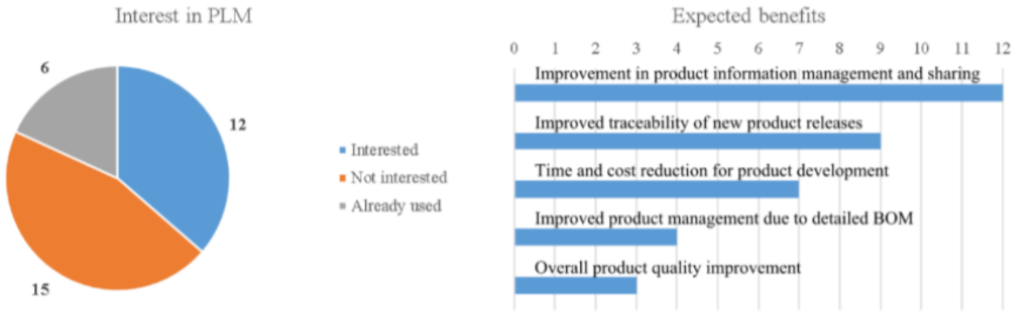


Figure 6Interest in PLM in by Italian companies in the piedmont Region (D'Antoni et al., 2017)

Six companies out of the 33 answered that they already have the PLM system, 12 showed their interest in deploying it and 15 showed no interest. Regarding the benefits, all the interested companies believe that PLM improves the management and sharing of product information, 9 believe that it improves the traceability of product new products and 7 believe that it reduces time and cost for product development. However, less than half of them expects that PLM improves product management due to detailed BOM and the overall product quality.

MES:

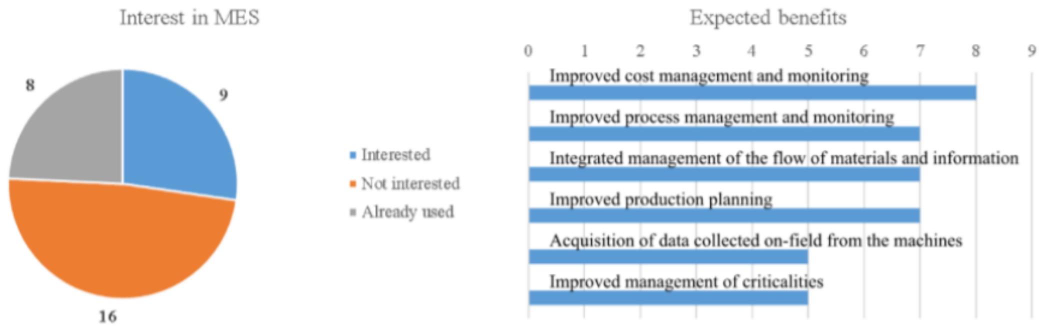


Figure 7 nterest in MES in by Italian companies in the piedmont Region (D'Antoni et al., 2017)

Regarding MES, 8 companies stated that they already have the system, 9 showed their interest and 8 were not interested. Among the interested companies, 8 expects an improvement in cost management and monitoring, 7 companies expect improvement in process management and monitoring as well as improved process planning and integrated management of the flow of materials and information. 5 believe that it improves management of criticalities and favors the acquisition of data from the shopfloor in real time.

PLM and MES:

Figure 8 Interest in PLM & MES integration in by Italian companies in the piedmont Region (D'Antoni et al., 2017)

As expected, none of the companies have yet integrated PLM and MES. However almost half the companies showed their interest in the integration of the two systems. This indicates the lack of enough awareness and the underestimation of the potential of the integration. 11 of the interested companies expects improved quality from such integration, 10 believe that it integrates design and production activities and 9 expect reduction in time to market and better tracing of product information

1.5 One-of-a-Kind Production (OKP)

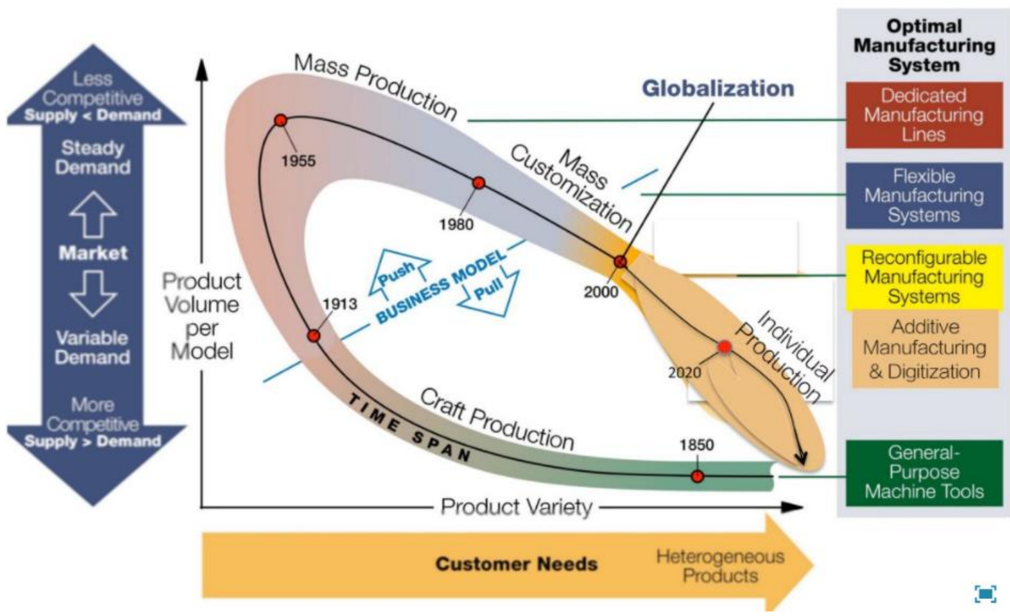


Figure 9 Evolution of manufacturing paradigms over the history

Before defining OKP and describing it, it is important to highlight the different trends that followed the evolution of different manufacturing paradigms over the history. The figure above summarizes in a simple way this evolution and considers product variety and product volume per model as main contributors in the distinction between different paradigms. Craft production is the most ancient way of manufacturing and it was characterized by high variety but very low volumes because of depending in human hands and very simple tools as mean of production. As time passed by, the increased demand pushed companies toward the adoption of more efficient manufacturing approaches to maximize production volumes and to reduce production costs. Several approaches were proposed but the most dominating method was the mass production which was developed in the early 1900s by the advent of assembly lines and is characterized by standardizing both the products and the manufacturing processes. This approach was successful for many years because despite the standardized products, the quality and the price were acceptable for customers. However, customer needs are known to be dynamic, and they evolve over time. Thus, their tendency to trade off customization with low prices has been progressively diminishing and they started to consider customization as one of their priorities. Mass customization was introduced in the late 1980s to respond to these challenges through flexible manufacturing systems and advance IT tools. Production volumes remains high in mass customization, but product variety is high as well. While MC represents the intermediate case of customization, OKP represents the extreme case where the production volume is very low, and the variety is extremely high and sometimes the company needs to produce a single product for only one company. This is big problem because it is almost impossible in OKP to apply same principles on which MP companies leverage on due to the low repetition of products features and manufacturing processes. OKP was originally involved in heavy industry such as ships production and power plant building (Tu & Dean, 2010), but it is currently extending to include almost every market.

OKP is known as customer driven approach because it focuses on manufacturing products according to individual customer requirements based on Engineer to Order (ETO) approach while aiming the same time to achieve the high efficiency and quality of mass production (Hong et al., 2010). (Wortman, 1991) expected that OKP is going to be the future of the European industry. As it can be seen in the following figure, the main difference between OKP and other approaches depends on the position customer decoupling point along the value chain. On the one extreme on the left, companies use Make to Stock (MTS) approach and the customer involvement is minimum as he places the order for a product already predefined and available in the market. OKP is customer driven approach because it is found on the other extreme where engineering and design activities do not start until the customer order is received.

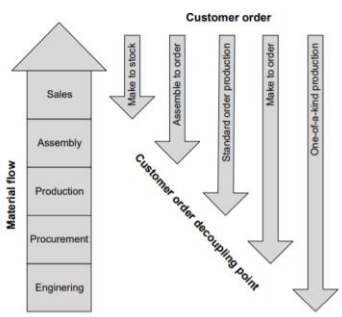


Figure 10 Position of Customer decoupling point along the value chain for different production typologies

The main characteristics of OKP industry can by summarized as follows (Tu et al., 2000)

- High customisation: The high customization results from the diversity of customer requirements which is associated with high uncertainty because each product has its unique characteristics. This level of customization is often associated with many reworks, high costs and longer product development and lead time.

- One go approach: Unlike batch or mass production where a prototype of the product is usually made to study the design and manufacturing processes before producing the final product, OKP companies must make the product in one go as it is not economically convenient to do otherwise. The one goes approach creates challenges as it is associated with high uncertainty in product design and process planning which lead often to additional costs and rework especially if the design or process planning are not done correctly. There is also a great challenge in identifying optimal production schedule and inventory plan.
- Loose production planning and higher inventory cost: The uncertainty of the product limits OKP companies in allocating their production resources because the processes are planned based on estimations that are often inaccurate.
- Continuous customer intervention: The customer requirements are clearly predefined and changes often in OKP regardless of the stage of the production. The earlier the customer asks for the intervention, the easier it is for the company to adapt the design to the requested changes. However, this is not always the case and OKP companies should be able to respond to these changes even if it is done at last stages of the production.
- Complicated data and information flow: The company produces different products at the same time and must manage individually the data relative to each product to avoid any conflicts that may lead to production mistakes such as assembling a part of a specific product to another one. This data is not only about product parts but is also related to process planning, design, and production schedule. Managing such data that flow simultaneously is real challenge especially when data of external companies is involved because OKP companies often order parts from subcontractors or suppliers.
- Complex logistics management: This problem is faced both inside and outside the organization. Good inventory handling and control is required within the company and complicated supply chain and distribution networks are required externally to manage the flow of materials and products from the suppliers and to the customers because orders are small while customers and suppliers are many and diverse.

Chapter 2: KNOWLEDGE MANAGEMENT

2.1 Overview of Knowledge Management

According to Resource Based View (RBV) a firm can maintain its competitive advantage if its resources are valuable, rare and inimitable. Among all the assets that an organization has, knowledge is considered as one of the most valuable assets that a firm can leverage on it to have a sustainable competitive advantage. This is because other assets and especially the tangible ones can be easily copied by other companies either because they are available in the market or because there is the possibility of reverse engineering. While for knowledge, it is so hard for competitors to imitate it and especially the implicit knowledge because it is embedded inside workers head and company's routine and skills. Even the company itself may struggle to codify its knowledge and transfer it between its different facilities ant there have been exhaustive studies by many companies to codify their knowledge by adopting different approaches.

However, having the knowledge itself is not sufficient if it is not managed wisely and effectively. This is because most companies accumulate experience and a lot of information and data during their life, but few can exploit these resources by lack of awareness about the importance of KM or because of not finding an effective tool to create a useful knowledge out of all that data.

KM however is not a new topic in the field of organizational and managerial sciences, and it has first emerged in the early 90s. Ever since, many articles covered KM topic as a discipline itself and the authors highlighted the role it plays in enhancing and managing the wealth and learning skills of an organization. Despite the presence of many articles about KM in the literature, no single definition was given either for knowledge or for KM. Davenport and Prusak, on their article about knowledge management, defined knowledge as a fluid mix of framed experience, values, contextual information, and expert insight that provides a framework for evaluating and incorporating new experiences and information. Holsapple (2005) related KM to human experience, interaction and interpretation of information. It was clear though that knowledge should not be confused with data or information and they should not be used interchangeably.

The table below provides a good demonstration of the difference between data, information, and knowledge where it considers data as a subset of knowledge and information, while information is a subset of knowledge.

Relationships		Definitions	Examples
K= I x U where	K= Knowledge I = Information U = Use	**Knowledge** *(Interiorized information put to action)* ⇑	I am in Paris today (*user context*) ⇓ I am going to wear a coat.
I = D x C where	I = Information D = Data C = Context	**Information** *(Data in context)* ⇑	The temperature is 10^0 Celsius today in Paris
		Data *(Raw facts)*	10^0 Celsius

Figure 11 Knowledge and Context Relationships

The most common way of classifying knowledge is dividing it into explicit and implicit knowledge. Explicit Knowledge is knowledge that is written, archived, expressed in words and numbers, and can be conveyed in specifications or manuals and can be used as learning materials and reference for others. This type of knowledge can be immediately passed from one individual to another in a formal and systematic manner because it already exists in a concrete form. This type of knowledge has the following characteristics:

- Tangible
- Visible knowledge
- Can be known publicly
- Can be accessed by many people

Tacit Knowledge is embedded in human capital in the form of know-how, experience, skills, understanding. This knowledge is very personal and difficult to formulate, making it very difficult to communicate or convey to others. Personal feelings, intuition, body language, physical experiences and rules of thumb are included in the type of tacit knowledge. The more tacit a knowledge is, the more valuable it is. It has the following characteristics:

- Intangible
- Invisible knowledge
- Private
- Only accessed by knowledge owners

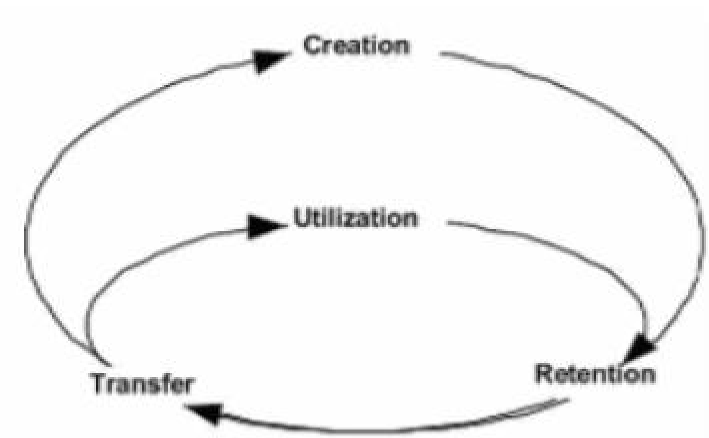


Figure 12 Different Stages of Knowledge Management

As showed above, knowledge management passes through 4 different stages: Knowledge creation, retention, transfer, and utilization.

1) Knowledge Creation: This step includes the activities of adding new knowledge. This occurs either when the firm finds a solution for a new problem or when it accumulates enough experience and information which can be useful for its activities.
2) Knowledge Retention: This step includes all the activities needed to keep the newly added knowledge in the system so that it can be used for future tasks.
3) Knowledge Transfer: It consists of the activities required to transfer knowledge between different persons and departments.
4) Knowledge Utilization: At this final step, the knowledge is applied to solve the business problem for which it was created.

2.2 Common Knowledge Reuse Approaches

2.2.1 Benefits of knowledge reuse in OKP

Before introducing the most common approaches in knowledge reuse, we must highlight the benefits of this practice in OKP filed. (Li et al., 2010) highlighted that previous knowledge plays an important role in creating value for OKP companies and previous successful products provide useful knowledge though the information generated during their production as well as the experience gained by the employees. The effective use of this knowledge and experience improves company ability in product development because it has the following benefits:

• Shortening lead time without trading off the high quality and decreases the unit product costs and saving significant amount of effort.

• Helping to avoid knowledge loss associated with product know-how as this kind of knowledge is embedded in designers' brain who may leave their job or get replaced.

• Enhancing problem solving skills and makes it faster since designers can refer to solutions for problem faced in the paced during the development similar products.

Thus, managing product knowledge are very important to maintain competitive position and this requires the creation of efficient knowledge base systems that can facilitate knowledge creation and reuse to exploit previous products knowledge.

2.2.2 Expert Systems

One of the widespread ways of knowledge management and reuse is the adoption of expert systems. These systems were introduced in the AI society in 1960 and they were used to resolve problems by using knowledge and measures rather than relying on human who usually use their experience in finding solutions. They then became famous tool to store, transfer and reuse the knowledge of manufacturing companies and had a crucial role in sustaining their competitiveness because they allow knowledge sharing at minimum cost and time. This is due to the limitation of human abilities in dealing with large amount of information which makes him less efficient in front of complex problems.

- Rule Based System:

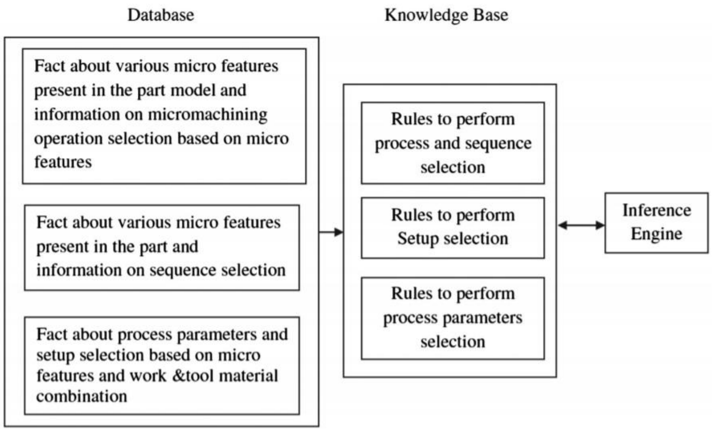


Figure 13 Architecture of Knowledge Rule Based Expert System

This kind of systems represents the knowledge as set of facts that represent information of the expert's rules are checked to execute certain tasks which is similar to their way in making decision. These rules are usually formed of two parts; the first part is the condition is preceded by if and the second part is the conclusion and is preceded by then. As shown in figure, it usually consists of the database that contains the information in terms of data or facts related to a specific domain, a knowledge base that contains the mathematical logic that describe the knowledge. Inference engine is responsible for interacting with user as it gets the input from him related to the problem he wants to solve, and it analyses it according to the logic inside the knowledge base in order to provide the solution as a form of output.

The way of reasoning can be either backward chaining or forward chaining. In backward chaining the process starts by the conclusion and checks if the rules related to it are true. Forward chaining works the other way around by going through the conditions and checking if they are true to arrive to the conclusion.

- **Case-Based Expert Systems**

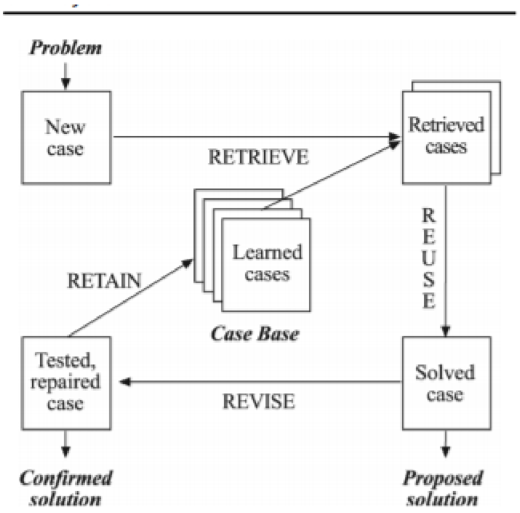


Figure 14 Case Base Reasoning Cycle

These are expert systems in which the case base is composed of past cases concerning the domain of interest. The search engine then elaborates a new solution from the past, by identifying previous solution to the most similar problem faced in the past and applying the same solution. This system is composed of 4 Re's (Aamodt & Plaza 1994) which are:

1. Retrieve the most similar case to the new case. In order to retrieve similar cases, each case is represented by a set of features that are compared one by one with those found in the Case Base in order to determine the overall similarity and detect the most similar case. The proposed formula obtained the similarity index by summing the weighted similarity index of each feature and dividing it by the summation of features weight.

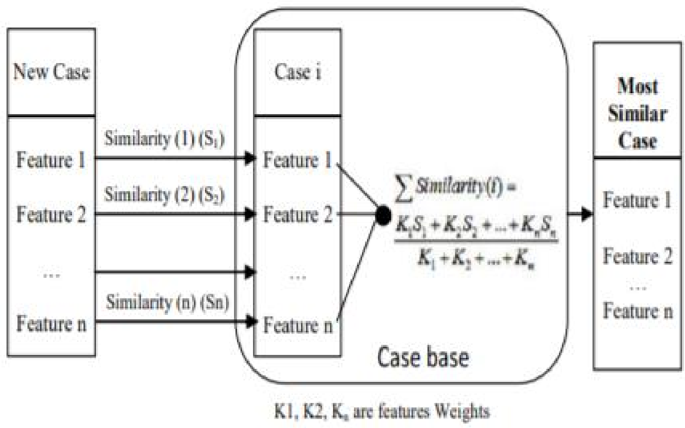


Figure 15 Similar cases retrieval within CBR expert system

2. Re-use the solution of similar case in the past to solve the problem of the new case

3. Revise the adequacy of the proposed solution in solving the problem.
4. Retain the solution if it can solve the new case.

This approach is similar to human's way of solving new problem by unconsciously trying to apply solutions for similar problems they faced in the past. In manufacturing, a designer or an engineer can follow the same approach by applying previous design solutions during the development of the new products if they share some similar features with previous ones. A new solution is created either if no similar cases are found or if the solution of similar cases is not found suitable to the new design problem. In this case, a new solution is created and assigned to the design issue and it is saved with previous cases. This approach is effective in retaining worker experience because it the knowledge base is being continuously updated with cases that express worker's experience. These cases are useful as hint for the designer who solved the problem himself in future design problems and it is also helpful for newly hired designers who can refer to similar cases in the past to build their experience.

2.2.3 Shape similarity approach

The different methods for retrieval of similar products can be divided into different groups according to the approach they adopt for detecting the similarity. (Li et al., 2004) stated that these approaches can be shape-based, knowledge based or ontology-based. Each method has its advantages and disadvantages, and its applicability and limitations depend on the context for which it is being applied. Shape similarity is one of the most common approaches in comparing products. This method is practical because the shape usually embodies meaningful knowledge related to design decisions, product functions and manufacturability. A company for example can know by looking at the shape of the products, the ability of its engineers in designing certain parts as well as the capabilities of its facilities in manufacturing certain products.

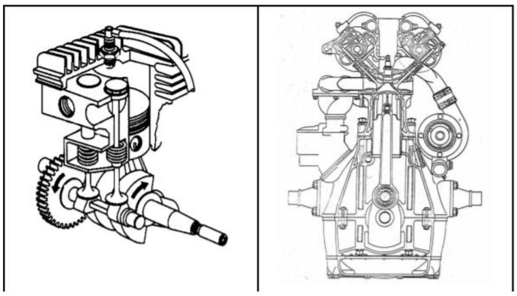


Figure 16 Single- and double-cylinder engines

If we look at the example in the figure above, we can notice the design of single- and double-cylinder engines which despite being two different types of engine may have the same design, manufacturing and quality control techniques because they only difference is in the number of cylinders and the size of the engine.

This approach is based on the topological and geometrical characteristics that determine the shape of the products which are represented through CAD models which can be either 2D or 3D. This approach assumes that the shape a certain product represents enough information to determine the degree of similarity it may share with other products. However, it overlooks other factors like material type, product functions, machinability of components and it's entirely focused on the shape.

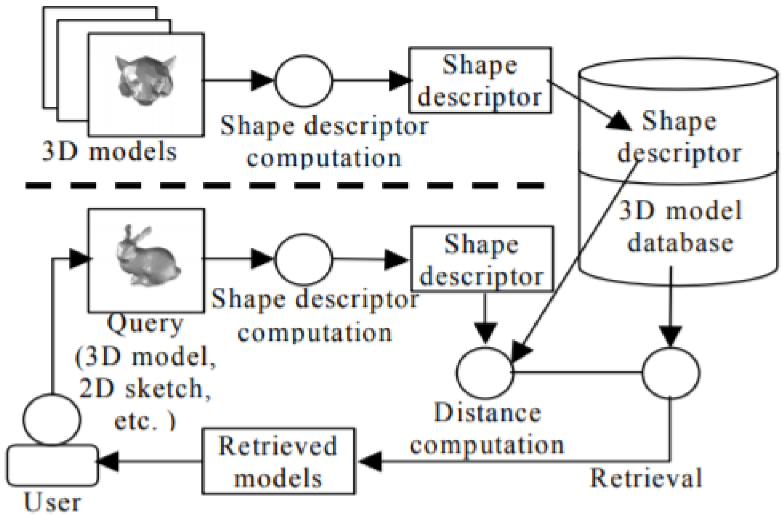


Figure 17 General diagram of 3D similar models search

Shape similarity is usually done in four steps which are illustrated in the figure above and they are the following (Ohubichi & Tsuyoshi 2003):

1. Query Insertion: At this step the user inserts the 2D or 3D model of the shape he wants to analyse.
2. Shape descriptor calculation: shape descriptors are extracted from the inserted model in order to represent model features by vectors in order to enable distance calculation in the next step.
3. Distance computation: This distance represents the dissimilarity between the two shapes, and it is zero when two shapes are identical and increases as difference increases.
4. Retrieval of similar shapes: Most similar shapes, which are supposed to have lowest dissimilarity distance, are retrieved from the database and displayed to the use

- Limitation of current similarity approaches:

There are many different methods and applications of shape similarity methods that can be found in the academic literature whose applications proved to be effective, accurate and fast retrieval of similar shapes. However, shape similarity is not always effective since there can be other contributing factors that reflect similarity between products. The figure below provides an example where important information can be lost if the designer depends only on shape similarity. The light valve principle which is used both by digital watch and smart window whose pictures are shown in figure. A designer may not use similar design considerations for the two products because he might easily overlook such similarity as windows and watches are considered two distinct products. This raises the need of more sophisticated similarity measure that goes beyond the shape of the product.

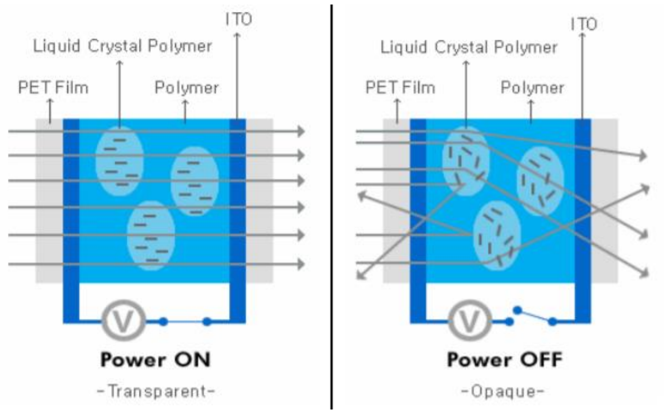


Figure 18 Light valve working principle

Figure 19 Images of smart watch and smart window

Chapter 3: SEMANTIC SIMILARITY IN KNOWLEDGE MANAGEMENT

3.1 Ontology

While it was originally linked to philosophy and referred to as a particular system of categories accounting for a certain vision of the world, in beginning of the 90s ontology has become increasingly widespread in computer science field such as AI, Computational linguistics and Database theory and applied in the fields of knowledge management, information retrieval and natural language processing. In AI field, an ontology is linked to an engineering artifact represented by a set of vocabulary used to describe it with their intended meaning. These vocabularies are connected through a hierarchy and related to each other through hyponym-hypernym relationship. In knowledge management field it is defined as a detailed explicit description of the concept. OWL is the common language usually used to model the knowledge by an ontology as it provides user with useful tools not only for the ontology creation, but the possibility of expanding or modifying it in an easy way.

According to (Young et al., 2007), ontology is considered as the base for sharing knowledge as it includes terms, properties, relationships among terms and semantic restrictions which is considered as an excellent method of representing knowledge in a formal way. It is also considered as the core and the backbone of most knowledge base systems (Kharbat & El-Ghalayini, 2008). This is due to its ability to solve semantic interoperability issue by integrating information that has different sources whose importance vanish if they are not represented in such a way that makes them provide useful knowledge. Therefore, ontology is considered highly important for manufacturing companies as it enables interoperability between different systems such as Product life cycle management (PLM), Enterprise Resource Planning (ERP), Supply Chain Management (SCM), Manufacturing Execution Systems (MES) and Supervisory Control and Data Acquisition (SCADA).

3.2 Semantic Similarity

According to the literature, semantic similarity, semantic relatedness, and semantic distance are terms that each has its own meaning even though the first two terms are used frequently synonymously. The between the two terms by stating that semantic similarity is narrower compared to semantic relatedness because the former refers to similar concepts with a synonym or an upper-class relationship such as in “car” and vehicle terms or “bank” and financial institution. The latter term however can refer to terms that have either informal relationship or even opposite expressions and still considered semantically related. “Tall” and “Short”, “Plane” and “Airport” an example of semantically related terms. The application of semantic similarity can be found in different fields such as Artificial Intelligence (AI), information retrieval and data processing (Baeza-Yates &Ribeiro-Neto, 1999).

3.3 Related Work on Semantic Similarity Measures

The literature focused three ways to determine the semantic similarity between objects in the ontology. The first approach represents an evaluation of the similarity based on conceptual distance and is called edge-based approach. The second approach proposes the evaluation of the similarity by the information content and is called the node-based approach. The third approach is a featured based approach that considers the degree of overlapping between sets of ontological features representing the concepts.

3.3.1 Edge Based Approach

This approach depends on edge counting to determine the distance between two concepts with the number of nodes representing the distance. It does not consider the information that the node itself has. The two main challenges with this approach were first to assign weights for arcs and or node in calculating the distance, and the other challenge was to distinguish between neighbour and descendant concepts while calculating the similarity.

- **Leacock and Chodorow formula:**

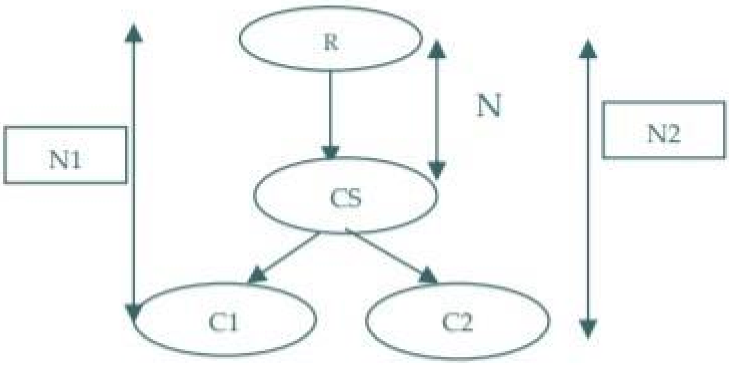


Figure 20 Sample Ontology

Consider the graph in figure, (Leacock et al., 1994) proposed this formula to compute the similarity based on edge counting approach:

$SIMlC(C1, C2) = -\log(\frac{\text{length}}{2.\text{D}})$

Where length is the shortest path length between C1 and C2 and D is the maximum depth in the hierarchy This way normalizes the distance between concepts by introducing a division factor that considers the maximum depth in the hierarchy.

- **Wu and palmer formula:**

Wu and Palmer (1994) define the conceptual similarity between a pair of concepts C1 and C2 as follows:

$$\text{simwp}(C1, C2) = \frac{2 * N3}{N1 + N2 + 2 * N3}$$

Where N1 represents the length measured by counting number of nodes in the path between C1 and C2, N2 is the length of path from C2 to C3, and N3 is the length of the path from C3 to the root of the hierarchy and by C3 we refer to the LCS (lowest common subsumer) of C1 and C2. The authors also found a formula for estimating the distance between the concepts which is the following:

$$Distwp(C1, C2) = 1 - \text{simwp}(C1, C2) = \frac{N1 + N2}{N1 + N2 + 2 * N3}$$

It is interesting to note that Wu and Palmer describe this metric relative to a taxonomy of verbs, but it can be applied equally well to other fields, as long as the concepts are organized in a hierarchy. One drawback however is that all arcs have same length and weight and the similarity between two concepts relies on the number of nodes connecting them. The other drawback of this method is that it may give higher similarity index between a concept and its neighbour and lower index with its descendent. This contradicts with the semantic logic because a descendant that makes a part of the concept should be more similar.

A small modification to solve this issue was proposed by (Slimani, et al., 2006), by adding one factor to the formula so that a similarity index is reduced for concepts belonging to different hierarchy. The formula is the following:

$$\text{SIMtbk}(C1, C2) = \frac{2.N}{N1 + N2} * PF(C1, C2)$$

As we notice, it is same as before, despite the penalization factor whose maximum value is one and this is the case when the two concepts belong to the same hierarchy and thy are not neighbours. However, if they are neighbours' concepts then the PF is estimated as follows:

$PF(C1,C2)=(1-\lambda)(Min(N1,N2)-N)+ \lambda(|N1-N2|+1)^{-1}$

N1 and N2 represent the same parameters discussed before while λ has a Boolean value which becomes 0 if two concepts are in the same hierarchy and 1 if they are not. Min (N1, N2) is obviously the minimum value between N1 and N2.

If we apply this formula to the same example, we hade before in the figure, we will have higher similarity between A and D than between A and B. This is because PF (A, D) is 1 and it will not reduce the similarity index computed by Wu and palmer formula. However it PF (A, B) is less than 1 and will decrease the similarity. To sum up, this method provides an effective way of solving the problem regarding neighbouring and descending concepts.

- **Shortest path Formula:**

While the previous method solved the problem of neighbouring and descendent concepts, it did not cover the problem of the weight each arc may have in the hierarchy. To solve this problem, (Abdelhadi, et al., 2017) proposed to assign weight for arcs in the hierarchy. It does not however use the same formula of Wu and Palmer, but it has a dedicated formula which was inspired by the original shortest path method proposed by (Rada et al.,1989) but it was extended to consider the wight of arcs instead of merely defining the distance based on the number of nodes. The following one is used to assign weight for arcs:

$$\mathrm{W}(m,n) = \left[\max\left(depth(m)\right) + \frac{N(n)}{NTNodes(G)+1} + 1\right]^{-1}$$

Where m and n are the two nodes directly connected, max(depth(m)) is the maximum depth of node. The depth is computed by attributing 0 value to the root nodes and 1 to the nodes that are connected directly to the root one. NT is the total number of nodes in the hierarchy and N(n) is the order number of the node n between their siblings

The second step after having calculated the weight is to determine the distance between the concepts and it is done according to the following formula:

$$\text{SDis (C1, C2)}= \mathrm{W}[SPath1] + \mathrm{W}[SPath2] + \mathrm{W}[CSPath]$$

Where, C1 and C2 are the two concepts being analysed, Spath1, Spath2, and CSPath represent the shortest path between C1, C2, LCS and the root node respectively and W [SPathi] is the distance of each path and it computed as follows:

$$\mathrm{W}[SPathi] = \sum_{j=1}^{k} Wj[m,n]$$

Where m and n are the two nodes which are directly connected in SPathi and k is the number of arcs in the Spath.

Lastly, we can compute the semantic similarity using this formula:

$$\mathrm{SimL(C1,C2)} = \frac{1}{\mathrm{deg} \times \mathrm{SDis(C1,C2)} + 1}$$

C1 and C2 are the concepts while deg is the impact of the degree of semantic distance on the similarity and it has always a positive value and should be less than one, however its value is defined based on experience. According to this approach the distance is the main determinant of the similarity because the similarity is inversely proportional to the distance and concepts are less similar when the distance is greater.

3.3.2 Information Content Approach

Where the previous approach considered the distance and the links between the concepts as a tool of finding semantic similarity, this approach considers the concept itself or the node and the information that the node has.

The measures developed by using this approach are the following:

- Resnik similarity formula:

Resnik (Resnik, 1999) was the first one to apply information content approach for identifying semantic similarity. According to this measure, the similarity between two concepts C1 and C2 is determined based on the information content found in the shared or parent concepts. The more information the two concepts share, the more similar they are. The formula is the following:

Sim_r (C1, C2) = -logP(mscs(C1,C2))

Where mscs is the most specific common subsumer and P (msc (C1, C2)) is the probability of its occurrence in a corpus.

$P(c)=\frac{freq(C)}{N}$

Where N is the total number of concepts and freq (C) is the frequency of concept occurrence in the hierarchy. This frequency of a concept also includes the occurrence of its hyponyms and as a result the probability should monotonically increases as we move up through the hierarchy and this in return decreases the information content for concepts at higher level.

- Lin's similarity approach:

(Lin, et al., 1993) followed a similar approach to that of Resnik with making slight improvements in results by overcoming the weak points that the original approach has. They included the information content of the individual concepts as well and not only that of the subsumes compared to the original formula.as follows:

$$\mathrm{SimL(C1, C2)} = \frac{2 * \mathrm{logP(mscs(C1, C2))}}{(\mathrm{logP(C1)} + \mathrm{logP(C2)})}$$

- **Jiang and Conrath's formula (1998):**

It is similar to the last two approaches but calculates the semantic distance rather than similarity, but it can also indicate the similarity level are because it decreases as concepts become more similar. This distance is the difference between the information content of the two concepts and the information content of their most specific common subsumer and is expressed in the following formula:

$Dist_{jc}$ (C1, C2)=2*logP(mscs(C1,C2))-(logP(C1)+logP(C2))

The recent approaches that worked on the refinement of these measures focused on the calculation of IC and not the similarity formula that are still the same. Most of these approaches focused on the role of hyponyms, hypernyms and some also considered depth to evaluated information content of concepts and with the aim of increasing the accuracy of the previous approaches. These variables can be defined as follows

- hyponyms(C): It is number of hyponyms of a concept that reflects the probability of occurrence of a concept inside the ontology. Hence, we can assume that concepts with higher number of hyponyms occurs more probably because these concepts can be expressed implicitly by means of all their hyponyms. It is important to mention that this number also includes the concepts itself in the count because evidently the concept C refers to itself explicitly. Following this logic, this number is always greater than zero and can be at least 1 even a leaf node placed at the bottom of the ontology and have no hyponym will still assign a number of 1. The generality of a concept is directly correlated with its number of hyponyms because it becomes more genera as this number increases and vice versa.

- hypernyms(C): This number is the opposite of hyponyms a concept with high number of hypernyms is considered more specific. It's because concepts with many hypernyms have less probability to occur. This number is useful because it helps to calculate more accurately the specificity of leaf concepts that have no hyponyms other than themselves. It is also useful to differentiate specificity concepts at the same depth because it considers the information of all the ancestor of the concept. Like hyponyms(C), this number cannot be zero because the concept itself is considered in the count and as a result, even the root node has hypernyms(root_node) =1

- depth(C): It represents the location of the concept with respect to the root node and it is calculated by counting number of links of different paths that connect the concept with the root node and considers the minimum one. It is equal to the number of hypernyms for concepts that with no multiple inheritance.

(Seco et al. 2004) proposed for the first time the calculation of information content based on the number of hyponyms and it is the following:

$$\text{IC seco et al.}\,(\mathrm{C}) = 1 - \frac{\log(\mathrm{hypo}(\mathrm{C}) + 1)}{\log(\mathrm{max\_nodes})}$$

where max_nodes is the total number of concepts in the hierarchy.

The main problem of this approach is that it results in same IC value for concepts having same number of hyponyms but that belong to different depth in the hierarchy. To tackle this problem, some recent approaches tried to make further improvements and proposed other models that considers some other factors such as the depth, number of leaves and number of hyponyms.

(Sánchez et al. 2011) proposed a model that considers the leaves of the concept instead of considering all the hyponyms. The model is the following:

$$\text{IC Sanchez et al. (C)} = 1 - \log\left(\frac{\frac{leaves(C)}{hypernyms(C)} + 1}{\max\_leaves + 1}\right)$$

Where max_leaves is the total number of leaves in the ontology and leaves(C) is the numbers of leaves of the concept.

(Meng et al. 2012) proposed another model that calculates IC of a concept based on its own depth and the depth of its hyponyms. In this way, each hyponym depth is taken in consideration where deepest concepts inside the hierarchy contributes more to the IC value. The formula is the following:

$$\text{IC Meng et al. (C)} = \frac{\log(depth(c))}{\log(\max\_depth)} \times \left(1 - \frac{\log\left(1 + \Sigma h \in \text{hypo(C)} \frac{1}{\text{depth(h)}}\right)}{\log(\max\_nodes)}\right)$$

Where h refers to hyponym of C and the depth is calculated by the following formula:

depth(C)= min_path (root, C)+1

Later, (Adhikari et al., 2015), extended the previous model and considered the contribution of hypernyms as well in calculating IC by multiplying the previous formula with the ratio of the normalized leaves number and hypernyms number as following:

$$\text{IC Adhukari et et al. (C)} = \frac{\log(depth(c))}{\log(\max\_depth)} \times$$

$$\left(1 - log\left(\frac{\frac{leaves(C) \times dhyper(C)}{\max\_leaves}}{\text{hypernyms(C)}}\right)\right) \times \left(1 - \frac{\log\left(1 + \Sigma h \in \text{hypo(C)} \frac{1}{\text{depth(h)}}\right)}{\log(\max\_nodes)}\right)$$

Where dhyper(c) is the number of direct hypernyms that are connected to C.

(Yuan et al., 2013) have also considered concept relative depth, number of leaves and that of hyponyms in calculating IC by using the following formula:

$IC_{Yuan\ et\ al.}(C) = fdepth(C)(1-fleaves(C)) + fhyper(C)$

Where $fdepth(C) = \frac{\log(depth(C))}{\log(\max\_depth)}$

$fleaves(C) = \frac{\log(leaves(C)+1)}{\log(1+\max\_leaves)}$

$fhyper(C) = \frac{\log(hyper(C)+1)}{\log(1+\max\_nodes)}$

3.3.3 Feature Based Approach

This method tries to tackle with the main problem of edge-based approach that the distance between concepts is uniform. Instead, it considered the feature overlap between concepts. Thus, the similarity depends on concept properties rather than the distance separating them. This approach was proposed based on Tversky (Tversky 1977) and the formula is the following:

$Sim_{tve}(C1, C2) = F(\Psi(C1) \cap \Psi(C2)) - \beta F(\Psi(C1) \setminus \Psi(C2)) - \gamma F(\Psi(C2) \setminus \Psi(C1)))$

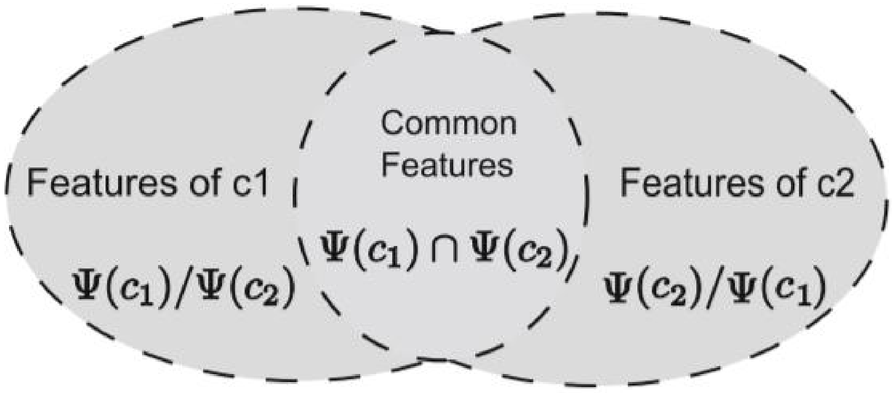


Figure 21 Tversky feature model (1977)

Where F is a function that represents the prominence of the various features of the two concepts and α, β and are the weight for each contributing component, Ψ (C1) and Ψ (C2) are the features of C1 and C2 respectively, Ψ (C1) ∩Ψ (C2) are the common features of the two concepts, Ψ (C1) \ Ψ (C2) are the features of C1 that remains after removing the common features shared with C2 and vice versa for Ψ (C2) \ Ψ (C1).

As it can be noticed, this approach considers both common and non-common features and similarity is obtained by subtracting non-common features from common ones.

The advantage of this method is its versatility as it remains applicable even when the concepts belong to different ontologies.

The limitation of this method is its dependence on weighting factors that can vary according to the ontology and the purpose of application. Another problem is that they consider that all features have the same weight despite that some features may contain more meaning than others and as a result may contribute more to calculating similarity.

Pirrò (2009), tried to exploit both the benefits of information content approach and feature based approach and solved the problem of adjusting the parameters in Tversky formula and he considered that the prominence of feature sets is reflected by the information content found in these features. Following his proposal Tversky formula transforms into this one:

Similar to Resnik metric, this one considers the information content of the least common subsumer of concepts, and this does not result in the maximum similarity value if concepts are identical. To solve this problem Pirrò proposed to assign similarity value of 1 for identical concepts and to use apply this formula if concepts are not identical. The following model is the following:

If C1 and C2 are identical: $SIM_{P\&S}$ (C1, C2)=1

Otherwise, $SIM_{P\&S}$ (C1, C2) = IC(msc(C1,C2)) - (IC(C1)- IC(msc(C1,C2))) – (IC(C2) - IC(msc(C1,C2)) = 3*IC(msca(C1,C2))-IC(C1)-IC(C2)

3.3.4 Accuracy of the proposed approaches

In evaluating the accuracy of different semantic similarity metrics, most of research used Rubenstein and Goodenough (1965) and Miller and Charles (1991) experiments as benchmarks. These experiments analysed similarity between different word pairs of English nouns with scale between 0 and 4 where 0 indicates that words are not semantically related and 4 indicated the strongest semantic relationship. Rubenstein and Goodenough performed the experiment on a group of 51 students who are native English speakers and gave them 65 pair of words to analyse. Miller and Charles performed a similar experiment but considered 30 pair of words instead of 65 and included 38 students. The correlation between the two results was 0.97 and the results of different semantic measurement metrices are evaluated by comparing them to this correlation. Pirrò repeated these experiments in 2009 but involved a larger group of people of 101 students that included non-native English speakers as well and obtained 0.97 correlation with Rubenstein and Goodenough and 0.95 with Miller and Charles. The high correlation obtained between different experiments emphasizes its reliability as de facto standard for evaluating metric accuracy.

Table 1 Semantic Similarity Experiments

Experiment	Year	Number of Pairs	Number of Participants
Rubenstein & Goodenough	1965	65	51
Miller & Charles	1991	30	38

P & S	2008	65	101

Among different papers that addressed the accuracy of semantic measures we can refer that of (Sánchez et al. 2012) where they made comparative analysis between all the above-mentioned methods showed that edge-based methods are the least accurate measures. Feature based methods provided more accurate results compared to edge-based method but they were less accurate than IC based methods and specifically those who compute IC based on the number of hyponyms

Chapter 4: KNOWLEDGE REUSE FRAMEWORK

The proposed framework is a similarity assessment method which is an important part of the KBS framework proposed by (Lombardi et al., 2019) which integrates PLM and MES as previously explained. Their framework focused on the architecture of the KBS model, the advantages of its application in OKP production and highlighted the role of identifying similar products. They did not however propose a method of identifying similar products and in this chapter, we aim to propose a similarity calculation method that is efficient and consistent with the model of KBS.

The application of similarity method within their framework provides more in-depth estimation of similarity between products than the previously discussed ones because as mentioned before, the shape-based methods alone ignore many other important parameters related to other information that describes the product. This is a big limitation especially for customized products, the shape similarity is not very common, and the degree of variability is high, and this urges the need of considering other factors which can contribute to similarity evaluation and provide useful help during product development. Keyword-based methods have also their limitations since they overlook the semantic meaning behind the words, and this becomes a big issue especially if information has different sources and different terms are used to describe similar parts.

The proposed method includes all the information related to the product from design until its ready for delivery. This increases the scope of similarity analysis and overcomes the limitations of other methods since it also identifies the similarity based on semantic meaning. Relevant amount of this information is included in PLM and MES systems because as mentioned in the first chapter they include important data related to the shopfloor and to the product itself during different stages of its life. PLM and MES integration are an effective tool for knowledge management by enabling the exchange of information between the two systems rather than functioning separately. The aim is to exploit in the best possible way the information of these systems to obtain the benefits mentioned in the previous chapters.

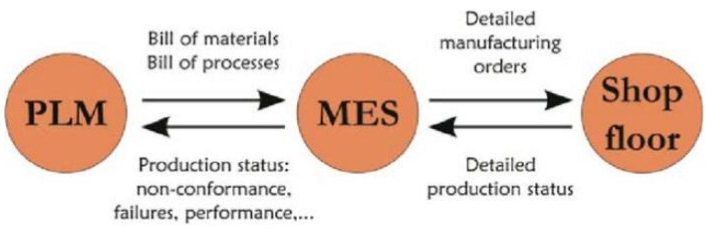


Figure 22 Data flow between PLM and MES (Bruno et al., 2020)

4.1 Knowledge Base System (KBS)

The system that provides the interface between MES and PLM is referred to as the knowledge-based System As is shown below, it acts as a bridge and it follows the following sequence: After a new order arrives the data related to this order is sent to the KBS to be compared with products that were produced in the past in order to identify the most similar ones. Not only this, the KBS also contains the relevant data related to shop floor such as processing time, production anomalies and other useful data. Thus, designers are not supposed to start everything from scratch but can get useful suggestions and reuse their previous knowledge without committing same errors they did in the past. Lastly the data related to the processing and production of the new order will be stored in the system and can provide useful information and tips for production of the future orders. Following this process, the KBS get enriched over time with accumulated data that can be used for future production activities.

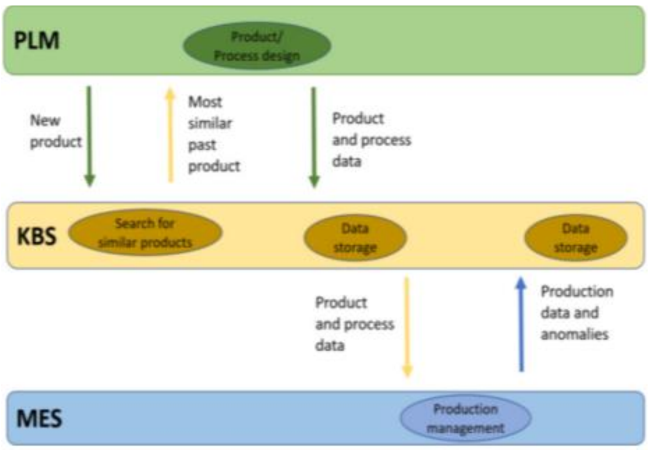


Figure 23 The flow of data from and into the Knowledge Base System (KBS), (Bruno et al. 2020)

4.2 Data model (Lombardi et al., 2019):

Before explaining the data model of the KBS, we should first identify the different data provided by the two systems which can be in the following tables together with their corresponding description.

4.2.1 PLM data

The following data is just a small a subset of all the data related to the product within its lifecycle. The table includes only the information related to the company resources and production process. This information is modelled using the available PLM software. In this case, ARAS Innovator was used and in the following table we can see the data and their corresponding description.

Table 2 PLM data and their description

Data type	Description
BOM.	The list names and quantities of all materials needed to realize the final product such raw materials, sub-assemblies and other parts.
CAD file	Contain the modelof the product
List of activities or bill of operations	The sequence of activities needed to obtain the produce the final product.
Activities description	Includes the location where the activity is supposed to take place, specifications and machines or equipment needed.
Check start	It includes the list of conditions to be satisfied before starting an activity. These conditions are related to the tool the status of the materials to be used
Check end	This check is mainly related to the product quality.
Machine description	It includes the type of machine to be used for each activity and its availability.
Family	Products are grouped in different families depending on their material type and production processes.
Subfamily	It's a subset of the family where products are divided according to their shape and function
Complexity	It's a parameter assigned to the product depending on the intensity of processing activity needed to achieve the final product.

4.2.2 MES data

The data related to the production collected by sensors and operators at the shop floor are explained in the following table:

Table 3 MES data and their description

Data type	Description
Check start results	It includes information about the satisfaction or not of the conditions necessary for starting the new activities.
Check end results	Includes information related to the occurrence of problems related to the production activities.
Machine failure	Includes information related to any possible failure of the machine during its activity.
Activity information	Includes information such as cycle time and set up time of activities performed.

4.2.3 KBS data model

The common database of the KBS is represented by entity-relationship model that include different entities that contain the data of PLM and MES which are mentioned in the previous tables and the links between them.

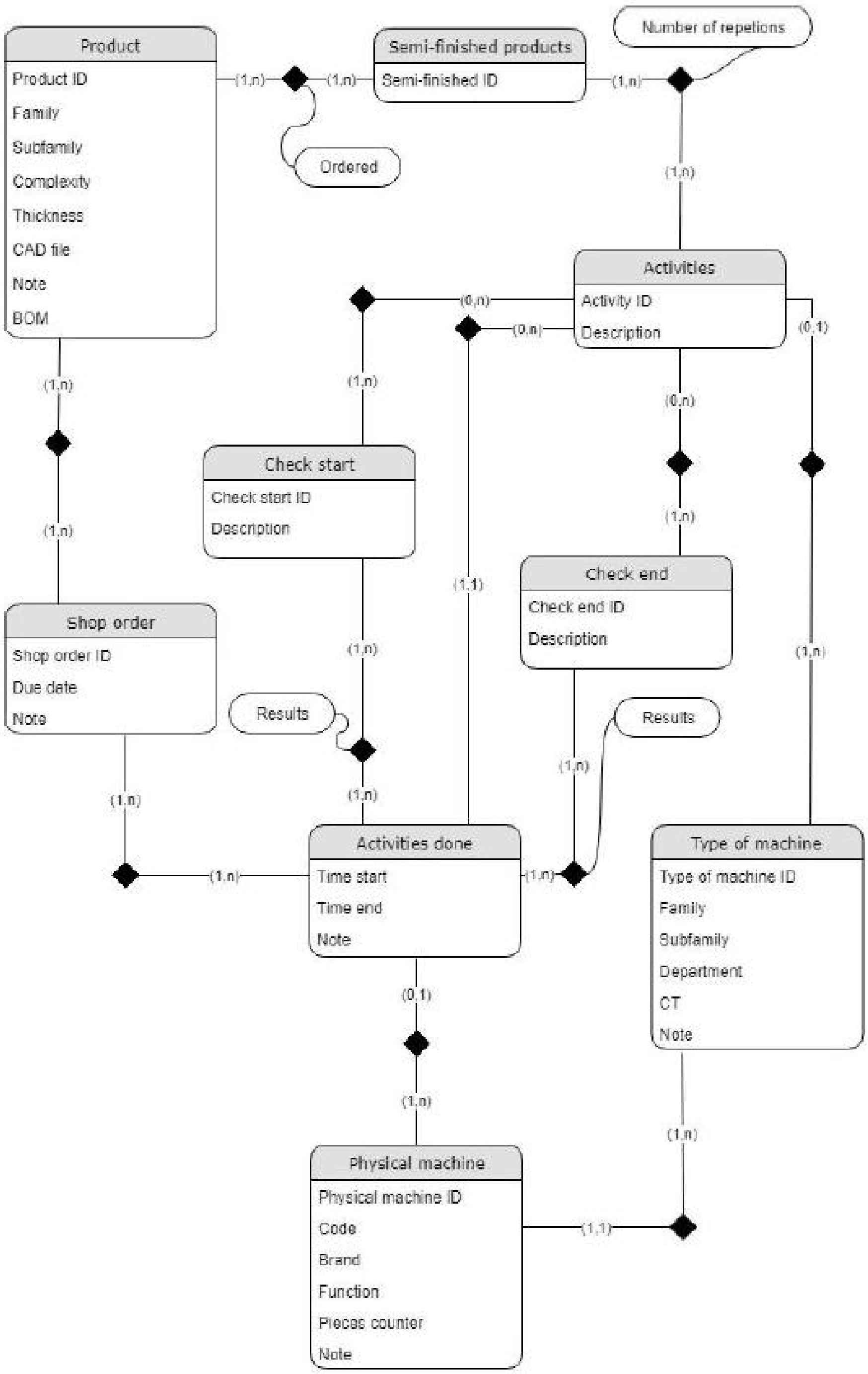


Figure 24 Entity-relatiosnip model of the common database (Lombardi et al. 2019)

4.3 Identifying similar products

KBS integrates data from different domains as it contains data related to PLM which as mentioned in the first chapter includes data related to all the stages of the product and at the same time it contains data collected from MES. These ontologies are made up of different classes that are connected through different relationships. As a result, a product can be represented by a graph that contain the specific classes that describe it. At the PLM level, the product graph can be made of classes representing attributes. At the same time, it has a graph corresponding to the different processing operations which are also represented by classes. The identification of similar products is determined based on the shared information between different products, and this is indicated by the overlap of their corresponding graphs. It is actually the same method applied by Bruno (2015) on manufacturing ontology but here we apply it on different ontologies..

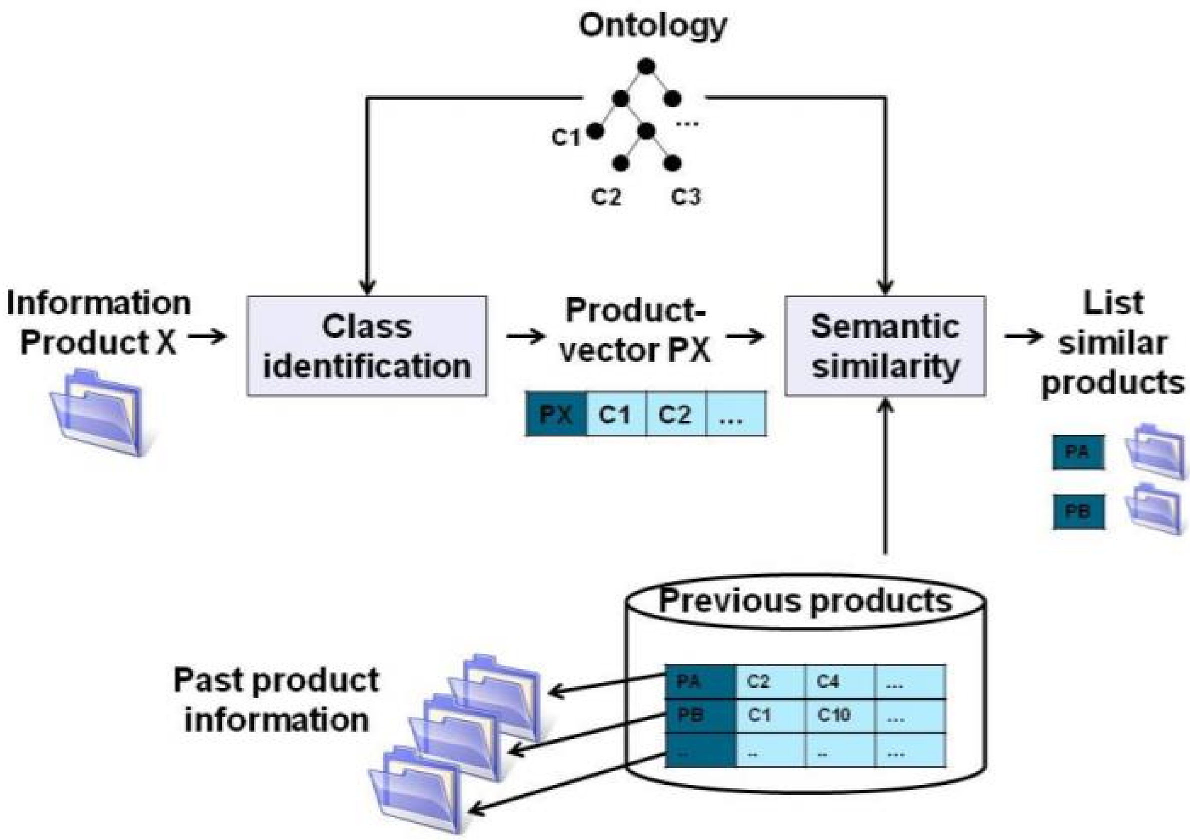


Figure 25 Similar products Retrieval using semantic measures (Bruno, 2015)

4.3.1 Information Content Calculation

As explained before, calculating information content of different classes in the hierarchy is the starting point before estimating similarity. We use the following IC formula to calculate information content:

$$\text{IC seco et al.}(\mathrm{C}) = 1 - \frac{\log(\mathrm{hypo}(\mathrm{C}) + 1)}{\log(\mathrm{max\_nodes})}$$

For the final estimation of semantic similarity of the overlapped graph is the following (Bruno, 2015): $Sim\ (a,b) \frac{\sum_{ci\in(Ga\cap \mathrm{Gb})} ICfamily\ ontology(ci)}{\sum_{cj\in(Ga)} ICfamily\ ontology(cj)}$

Where c is the concept belonging to product graph. Ga and Gb graphs of products a and b respectively and $Ga \cap \mathrm{Gb}$ is the overlap between the two products' graph.

4.4 Product Ontologies

The sample ontologies illustrated in this section belong to the products of car prototypes manufacturing company that is presented in the last chapter. However, the study is based only on two ontologies representing final product and material hierarchy due to the availability of this information from the company. Despite not including other domain ontologies such as manufacturing ontology, the main goal is not affected because the same similarity method applied to these two ontologies can be applied when all ontologies are included. Thus, the approach does not change but the results become more insightful and meaningful because all domains will be included.

4.4.1 Final product ontology

Final product ontology represents the hierarchy of product family, subfamily and complexity. Protégé was used to make represent the hierarchy in different classes. The root class is thing and then it has one subclass which is the final product which is furtherly divided into subclasses representing product families. The classification of these families based on the production characteristics because for example panelling requires highly deformable materials and for this reason panelling components are grouped together. The same criterion applies for other families, as structural elements need to be stiff and strong enough to support provide structural support to the product. The families are the following: panelling, structural elements, other artifacts and ballast and non-structural elements. Each of product family classes is made up of different classes representing product subfamilies which provide more detailed description of the component and is based on its function. Product with same subfamily have similar production cycles. Lastly the complexity class which is the subclass of subfamily is placed at the bottom of the hierarchy and has no further subclasses. This class is associated with the length of the processing cycle and the quality of the product and can have a value between 1,2 and 3 where 3 indicates that the quality is high and consequently the processing time is long. Figure 27 shows the first two levels in final product hierarchy that include product family in one level and subfamily in a lower level Figure 28 shows also the bottom level classes of panelling class that include complexity. It is important to highlight however that all of product classes has the same deepness and include three level but here we show only panelling for simplicity.

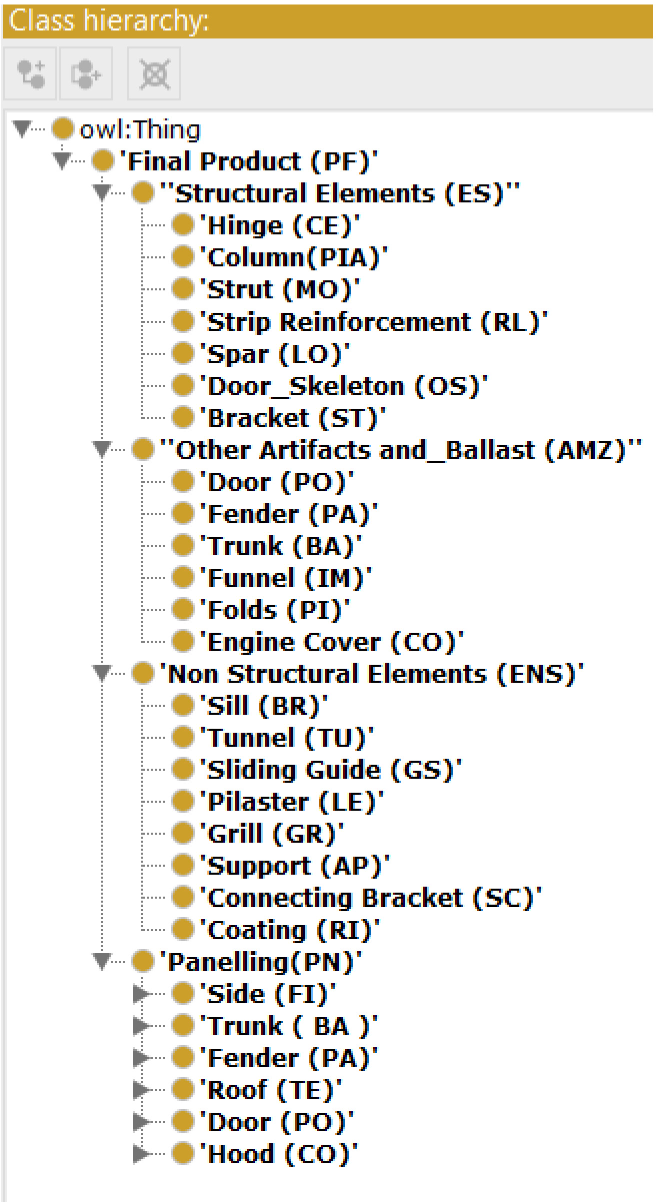


Figure 26 Screenshot of first two levels of final product ontology developed using PROTÉGÉ

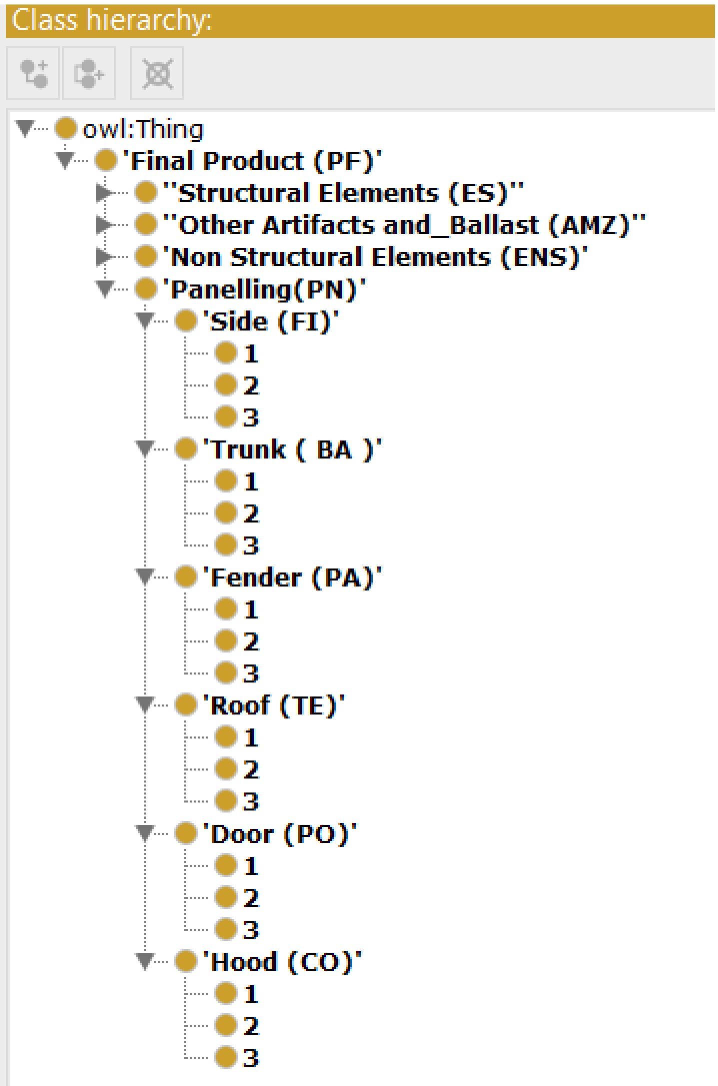


Figure 27 Screenshot of the second and third levels of panelling family within final product ontology developed using PROTÉGÉ

4.4.2 Material ontology

The second ontology corresponds to the hierarchy of materials classification used in the production and their corresponding base materials, surface state and surface finish. The five material types used inside the company and they are the following classes: Aluminium, non-galvanized metal sheet, hot dipped galvanized metal sheet, electrogalvanized sheet and Zinc-Magnesium-Sheet. Each of these subclasses is furtherly extended to include base material, surface state and surface finish classes represented by abbreviations corresponding to them. The first figure below shows the first two levels of the material ontology. The first only four class represents the material type. Base material, surface finish and surface state are at the same level and represent the subclasses of each material type. The second figure shows the classes of the bottom level for aluminium material. Obviously, the other materials have also their subclasses at the bottom level but we are not showing them in the figure, however they are shown in the appendix 2.

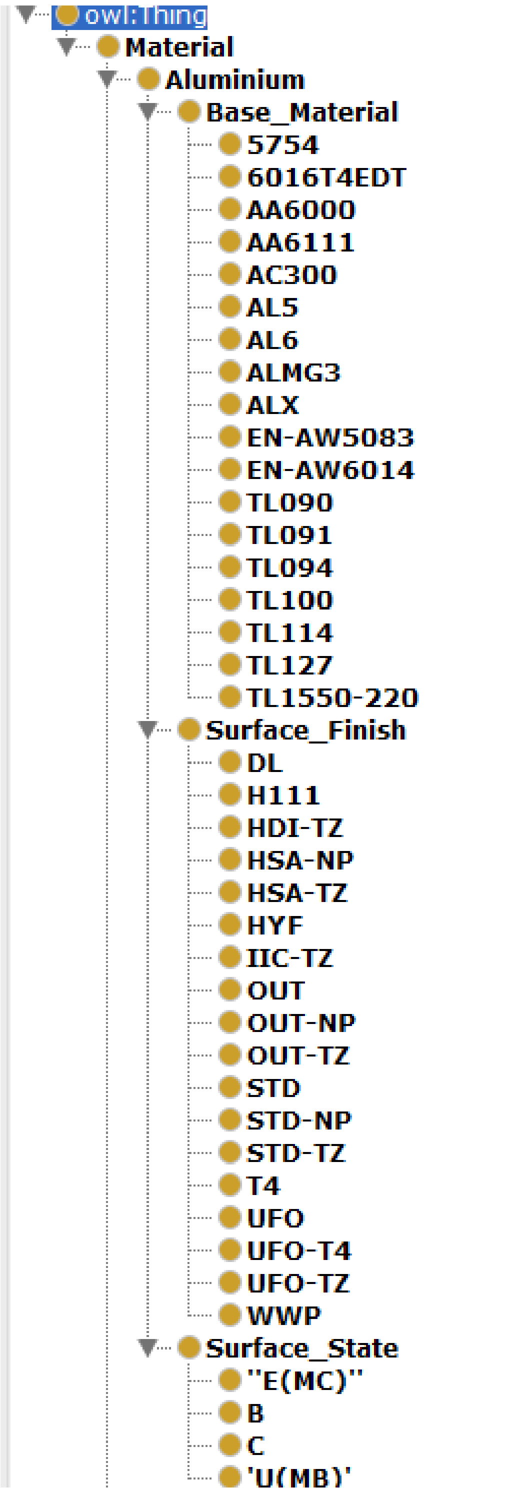


Figure 29 Screenshot showing all subclasses of Aluminium within material ontology developed using PROTÉGÉ

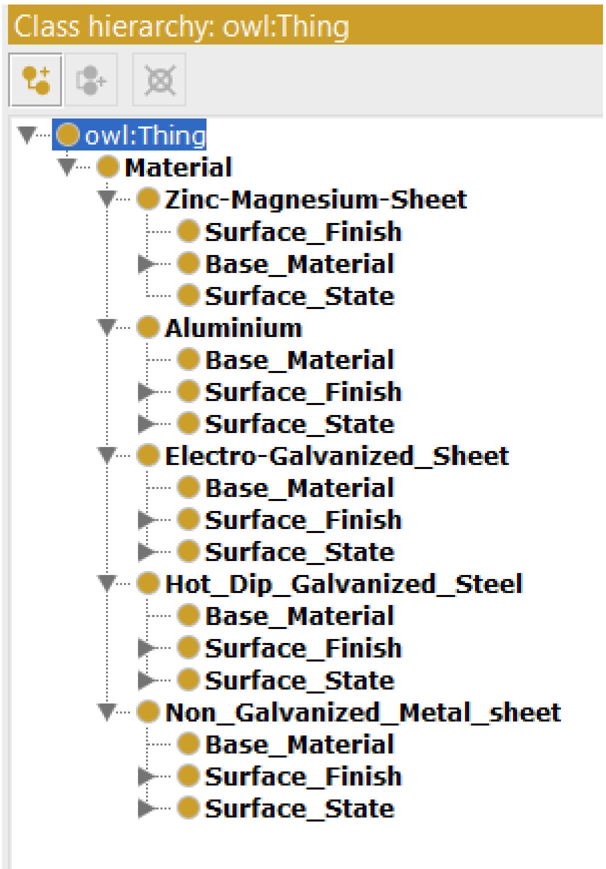


Figure 28 Screenshot of first two levels of material ontology hierarchy developed using PROTÉGÉ

Chapter 5: COMPANY USE CASE

5.1 Company Profile

The company of our use case is Eurodies, located in Avigliana in Italy. It is a supplier of key automotive components for prototypes of famous car manufacturers around the world such as Alfa Romeo, Audi, BMW, FCA, Ford, Lamborghini, Mercedes, Mini, Opel, Porsche, Rolls Royce, Seat, Skoda, and Volkswagen. This sector is characterised by high customization since the company must deal with individual customers representing different brands and models and the level of similarity is too low. Moreover, the production volume is not high and sometimes the company needs to produce only one product, and this makes it a good example of OKP company. The company faces design variations which lead to manufacturing variations and increases complexity to identify the optimal design and manufacturing process respecting quality, time and cost.

5.2 Production Process

As illustrated in the figure below, the process starts after the company receives a new production order including the CAD models of the components. These models are then received by the designers at the technical office where the dies needed for the production and production processes are defined. Following dies production, metal sheets are transferred to the Laser office where they undergo trimming process using a two-dimensional laser. Once the trimming process ends and the metal sheets obtain the desired shape, they are sent to the pressing area where they are pressed to obtain the semi-finished shape. As a last step, they are sent back to the Laser office where 3D lasers are used to obtain the final shape.

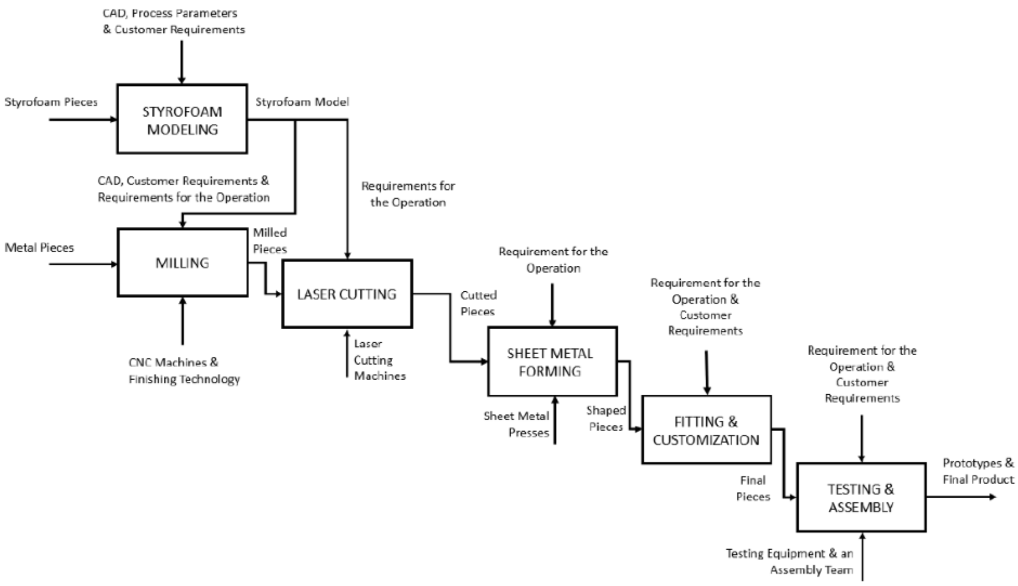


Figure 30 IDEF0 of production process at Eurodies company (Bruno at al., 2020)

5.3 The need of knowledge digitization

The nature of OKP puts the company under the pressure that necessitate the optimal exploitation of its knowledge to reduce the time to market and to avoid high material waste during production. They key solution is to leverage on the knowledge of the previous products to learn from previous experience and avoid their reoccurrence with new products. Although the company tries to exploit this knowledge leveraging on its employees' experience through the embedded knowledge in their minds, it is not sufficient survive the increased competition especially with the continuous innovation pursued by other actors in the market.

This traditional way of knowledge management adopted by the company through its reliance on its employees causes the improper exploitation of its accumulated knowledge gained through learning from producing previous components and sometimes it causes the loss of important knowledge. Two problems arise here, the first one is related to the level of accuracy of these estimations and the long time a normal human take to analyse the new order and compare it with previous ones. This is because no matter how much employees are skilled; they still take time to analyse all the characteristics of the new product and they are human beings who by their nature are prone to making mistakes who unfortunately might come at a very high cost. The second problem is the inability of the company to transfer knowledge embedded inside experienced workers head to new hired ones or even to transfer it to another facility if in the future it will decide to open a new production facility due to a possibly increased demand.

5.4 Role of Similarity Framework

Finding similar products as explained in the previous chapter can play a crucial role in the reuse of the industrial knowledge through the exploitation of data of both MES and PLM. of the bidirectional flow of data is giving technicians responsible for product development access to process information, stored in the KBS, so that they can make better design decisions. In this way, analysts and designers become aware with products that faced anomalies during production with a full description of that problem. As a result, company keeps track of historical production issues and avoid the repetition of previous problems in future production.

Data from PLM are related to CAD files, BOM, list of activities and their description, check start and check end, machine description as well as family, subfamily, and complexity of the product. MES on the other hand provide shop floor related data generated during production such as cycle time, the machine set up, the time waiting for the operator and machine failures. This data is helpful in defining production cycle times for new products.

Finding similar products in an automatic way creates a great advantage with respect to the manual way. This solution should be able to list similar products within a few seconds and provide them to the user. This saves the search time which was usually required by the employee to find similar items. It also reduces the time needed to define production cycle for new orders and increase the company's overall efficiency and responsiveness to market demands. The reason is that by this system employees no longer need to treat each new product as a completely different one and start everything from scratch because they can identify main differences and apply necessary changes for production cycles already defined form similar products. The similarity method discussed in the previous chapter is applied here to classify similar products in the company's database.

The implementation does not fully exploit all product related data because the application is done only on two ontologies related to PLM. One of them includes product family, subfamily, and complexity and the other one contains material type, base material, surface finish and surface state.

5.5 Identifying similar products

5.5.1 Information Content of final product and material ontology

In our application, we try to use semantic similarity approach on data coming from PLM as the company is already using Aras software to digitize product related information. Among the available information, we have those related to final product and material classification. In the similarity approach that we proposed in the previous chapter, similarity analysis starts by calculating information content of different classes in the hierarchy. This is because we determine similarity based on the amount of shared information between products: Different classes contribute differently to the comparison of products because common specific attributes are more meaningful than generic ones. For example, if two products only belong to the same family, this shows low similarity because each family is composed of a specific number of subfamilies and each subfamily can have three different complexity levels. Thus, family is generic while subfamily is more specific, and complexity is the most specific among all. This approach is consistent with information content methodology in identifying semantic similarity as it considers leave nodes as the most informative and specific while those are the less informative and the most generic.

If we apply Pirró and Seco IC formula, explained in previous chapter, on the classes of final product and material ontologies we obtain the tables that contain information content of each class in the two ontologies. The following table shows the information content of final product family classes where an abbreviation beside each class name is given corresponding to the first letters of the term in Italian language and we can see the original file in appendix 1. Information Content for classes of material ontology are shown in appendix 2.

Table 4 Information Content of final product ontology classes

Class	DESCENDANTS (desc)	IC = 1-log(desc+1)/log(totdsec)	Family Class	IC	Subfamily Class	IC	Complexity Class	IC
PF	112	0	**Panelling (PN)**	0,32	**Side (FI)**	0.71	1	1
							2	1
							3	1

					Trunk (BA)	0.71	1	1
							2	1
							3	1
					Fender (PA)	0.71	1	1
							2	1
							3	1
					Roof (TE)	0.71	1	1
							2	1
							3	1
					Door (PO)	0.71	1	1
							2	1
							3	1
					Hood (CO)	0.71	1	1
							2	1
							3	1
			Other Artifacts andd_Ballast (AMZ)	0.32	**Engine Cover (CO)**	0.71	1	1
							2	1
							3	1
					Folds (PI)	0.71	1	1
							2	1
							3	1
					Funnel (IM)	0.71	1	1
							2	1
							3	1
					Trunk (BA)	0.71	1	1
							2	1
							3	1
					Fender (PA)	0.71	1	1
							2	1
							3	1
					Door (PO)	0.71	1	1
							2	1
							3	1
			Structural Elements(ES)	0.29	**Bracket (ST)**	0.71	1	1
							2	1
							3	1
					Door_Skeleton (OS)	071 .	1	1
							2	1
							3	1
					Spar (LO)	0.71	1	1

							2	1
							3	1
					Strip Reinforcement (RL)	0.71	1	1
							2	1
							3	1
					Strut (MO)	0.71	1	1
							2	1
							3	1
					Column(PIA)	0.71	1	1
							2	1
							3	1
					Hinge (CE)	0.71	1	1
							2	1
							3	1
			Non Structural Elements (ENS)	0.26	**Sill (BR)**	0.71	1	1
							2	1
							3	1
					Tunnel (TU)	0.71	1	1
							2	1
							3	1
					Sliding Guide (GS)	0.71	1	1
							2	1
							3	1
					Pilaster (LE)	0.71	1	1
							2	1
							3	1
					Grill (GR)	0.71	1	1
							2	1
							3	1
					Support (AP)	0.71	1	1
							2	1
							3	1
					Connecting Bracket (SC)	0.71	1	1
							2	1
							3	1
					Coating (RI)	0.71	1	1
							2	1
							3	1

5.5.2 Calculating similarity:

The following table represents the available products and their attributes. On this table we removed some of the original attributes provided to use by the company like Zone, CAD, dimensions, etc. This is done on purpose because the only important ones are those included in the two ontologies.

1Dataset of different products from Eurodies Database

Table 5 Products List and part of their attributes

Prod-uct id.	Material Type	Base Material	Surface-Finish	Surface state	Family	Subfamily	Complexity
6000	E	CR290Y490T-DP	GI50/50	O	Panelling (PN)	Roof (TE)	3
5014SX	A	EN-AW6008	STD-NP	E	Panelling (PN)	Side (FI)	2
5014DX	A	EN-AW6008	STD-NP	E	Panelling (PN)	Side (FI)	2
4721	A	EN-AW6008	STD-NP	E	Panelling (PN)	Side (FI)	1
4340	A	EN-AW6008	GI40/140	PZ	Panelling (PN)	Side (FI)	2
4884	A	EN-AW6008	Uncoated	E	Panelling (PN)	Side (FI)	2
4953	G	CR210	Uncoated	P-O	Panelling (PN)	Roof (TE)	2
F7281	A	CR290Y490T+DP	GI40/140	E	Other Artifacts and_Ballast(AMZ)	Trunk (BA)	2
P2918	A	EN-AW6008	OUT	B	Other Artifacts and_Ballast(AMZ	Engine Cover (CO)	1
O2819	A	EN-AW6008	STD	E	Structural Elements(ES)	Column (PIA)	2
I27819	G	HX180	Uncoated	E	Non Structural Elements (ENS)	Grill (GR)	3
F2814	G	CR210	Uncoated	U	Structural Elements(ES)	Strut (MO)	2
N1823	G	CR210	Uncoated	U	Panelling (PN)	Roof (TE)	1
J1829	Z	DX56	GI70/70	O	Other Artifacts and_Ballast(AMZ)	Engine Cover (CO)	2
I1213	Z	CR290Y490T+DP	GI60/60	P-O	Other Artifacts and_Ballast(AMZ)	Folds (PI)	1
Q1283	Z	EN-AW6008	GI40/40	O	Panelling (PN)	Door (PO)	1
Y2819	E	EN-AW6008	ZE40	O	Structural Elements(ES)	Hinge (CE)	2
H1829	E	TL094B	ZE40	P-O	Structural Elements(ES)	Bracket (ST)	3
JS194	E	CR380LA	ZE75	E	Non Structural Elements (ENS)	Sliding Guide (GS)	3
D9104	M	DX51D	ZM35/35	U	Non Structural Elements (ENS)	Grill(GR)	2
M9010	M	DX52D	ZM40/40	U	Other Artifacts and_Ballast(AMZ)	Door (PO)	1

U3819	M	DX51D	ZM35/35	E	Structural Elements(ES)	Ginge (CE)	2
F9201	Z	EN-AW6008	GI60/60	E	Non Structural Elements (ENS)	Sliding Guide (GS)	2
N1940	E	EN-AW6008	GI50/50	O	Panelling (PN)	Door (PO)	3
O3918	G	CR210	Uncoated	O	Structural Elements(ES)	Door Skeleton (OS)	3
K1039	G	HX180	Uncoated	E	Panelling (PN)	Side (FI)	2
T3910	G	HX250	Uncoated	E	Structural Elements(ES)	Column(PIA)	1
L2104	A	CR290Y490T+DP	OUT	E	Panelling (PN)	Fender (PA)	2
K2930	M	DX51D	ZM35/35	U	Non Structural Elements (ENS)	Grill (GR)	2
NEW	E	CR290Y490T-DP	GI50/50	O	Panelling (PN)	Roof (TE)	1

As highlighted in chapter 4, for our analysis, we are considering only product family, subfamily, complexity, material type, base material, surface finish and surface state for calculating the similarity but analysis can extend to include other attributes following the same logic. Thus, the contribution of other attributes in similarity is not considered.

To calculate the semantic similarity, we use formula introduced in the previous chapter and we add new factor W because similarity comes from two ontologies. It determines the contribution of each ontology to the similarity. This task can be typically performed by the technician who are fully aware of which ontology can contribute more and indicate more meaning. In our case we can give it a random number just to show how the framework works and because we are not informed enough to give a fair estimation W value.

$$\text{Sim(Pa,Pb)} = W \times \left(\frac{\sum_{ci \in (Ga \cap Gb)} IC family\ ontology(ci)}{\sum_{cj \in (Ga)} IC family\ ontology(cj)} \right) + (1 - W) \left(\frac{\sum_{ci \in (Ga \cap Gb)} IC family\ ontology(ci)}{\sum_{cj \in (Ga)} IC family\ ontology(cj)} \right)$$

Products are represented by vectors that can be represented by a graph composed of the concepts that correspond to the product attributes. This can be shown in the following figure:

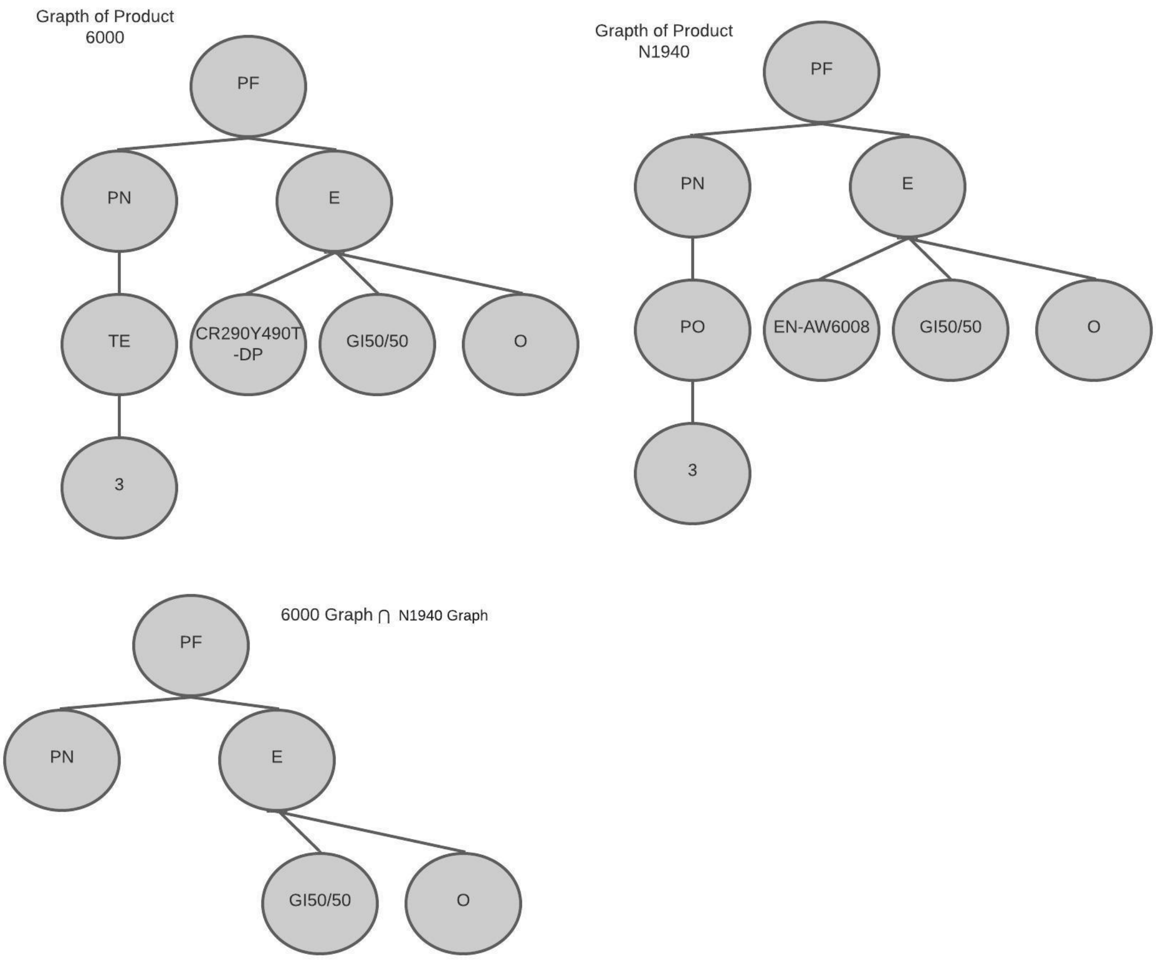


Figure 31 Products Graph Representation and their overlapped graph

5.6 Implementation

5.6.1 Flow chart of similarity process

- After a new product arrives and is inserted in the database. The user can start similarity search by inserting the new product id and the weight he wants to assign to semantic index derived from product family ontology that will be derived from each ontology.If the inserted id is wrong, a message is shown to the user and asks him to check the inserted id. Once the product is found, the first iteration starts through the columns in the database corresponing to products attributes in order to extract the attribiutes of the inserted product by the user. These attributes are, family, subfamily, complexity, material type, base material, surface finish and surface state.
- After the attributes are extracted, the information content (IC) of the classes corresponding to these attributes are added in order to obtain the total IC of chosen product. For each product, ther there are two ICs, one belongs to classes of final product ontology and the other one belongs to the classes coming from material ontology. This means that $IC_{\text{final product ontology}}=\sum IC\ final\ product\ classes$ and $IC_{\text{material ontology}}=\sum IC\ material\ ontology\ subclasses$

- Following the extraction of selected product attributes and the estimation of both of its ICs, the comparision iteration starts. It starts by going through the remaining products in the database, one by one and compares each of their attributes with those of the selected product. If a match in attributes is found, the IC of the class corresponding to it is added to the common IC. Also in this case we have two common ICs correponfing to each ontology.
- At the end of comparision stage, we obtain IC common material = ∑IC common material ontology classes and IC common final product ontology =∑IC common final product classes
- At the end of iteration through each prouct(i) ≠ selected product, the common ICs and those of the selected product are added to a tuple.
- Once all the iterations are done, the final similarity index for each product is obtained using this formula:

Similarity Index (SI) $=W\times\left(\frac{=\sum IC\ common\ final\ product\ classes}{\sum IC\ final\ product\ classes}\right)+(1-W)\left(\frac{\sum IC\ common\ material\ ontology\ cllasses}{\sum IC\ material\ ontology\ lasses\ of\ selected\ product}\right)$, following SI calculation, the products are grouped inside a dictionary with product id as Key and SI as value

- Lastly, the products inside the dictionary are sorted according to the SI in decreasing order and the 10 most similar products are shortlisted.
- The final output is a table that shows the user the most similar products and their corresponding similarity index.

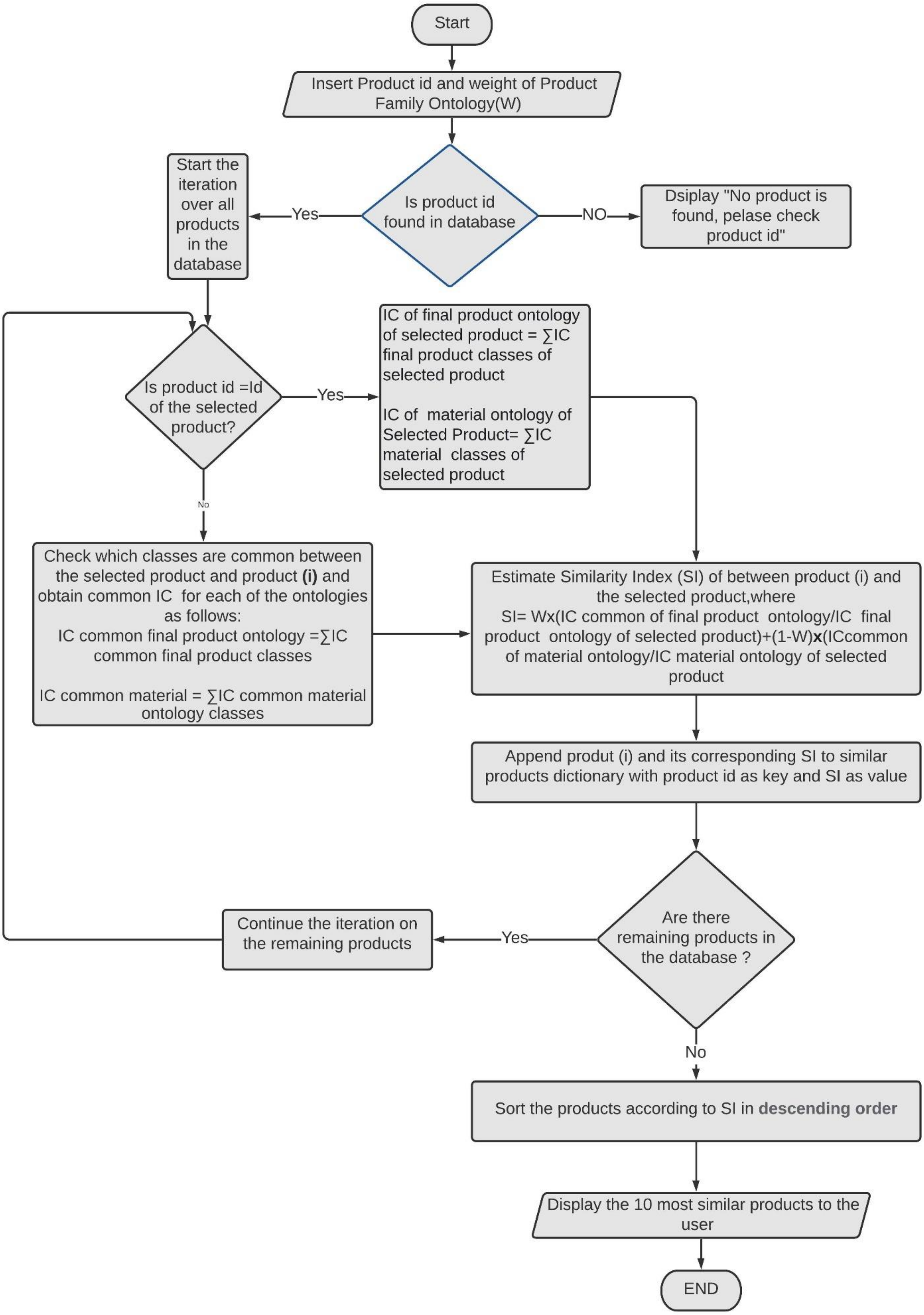


Figure 32 Flow chart of similarity calculation process

5.6.2 Code Explanation

Python was the programming language used for writing the code as well as the user interface.
Opepyxl library was imported to access the excel file directly in python application and to perform the iteration over different rows representing products and columns representing their attributes.
Tkinter library was used to create graphical user interface.
Itemgetter is imported to be used in the sorting function

```
from operator import itemgetter
from tkinter import StringVar, Tk, Label, Entry, Button, ttk, messagebox
from openpyxl import load_workbook
```

Each raw of the excel file associated with a product and each column represents the corresponding attribute such as number, material, etc. The corresponding attributes for row indices in the original excel file that contains products list we are calling in our code are the following:

Explanation of raw Indices meanings inside the code

row [0]	Product id
row [6]	Material Type
row [7]	Base Material
row [8]	Surface Finish
Row [9]	Surface State
row [11]	Family
row[11]-raw [15]	Subfamily
row [17]	Complexity

```
#Information Content for Material Ontology
material_a_ic = 0.40958
material_m_ic = 0.23621
material_g_ic = 0.238838
material_z_ic = 0.22734
material_e_ic = 0.231081
base_material_ic = 1
surface_finish_ic = 1
surface_state_ic = 1
```

The first step was to define the information content of each concept as constants before using them in different functions

```
class Semantic_simlarity_calculator:

    def __init__(self):
        self.root = Tk()
        self.textEntry = StringVar()
        self.label = Label(self.root, text="Find Similar products", font="Verdana 18 
bold").grid(row=1, column=1, padx=10,pady=10)
        self.label1 = Label(self.root, text="Insert product id ", font="Verdana 14 
bold").grid(row=2, column=1, padx=10, pady=10)
        self.entry = Entry(self.root, bd=2, width=20, font="calibri 20 bold")
        self.label2 = Label(self.root, text="Insert W value", font="Verdana 14 
bold").grid(row=3, column=1, padx=10, pady=10)
        self.entry1 = Entry(self.root, bd=2, width=20, font="calibri 20 bold")
        self.entry.grid(row=2, column=2, padx=10, pady=10)
        self.entry1.grid(row=3, column=2, padx=10, pady=10)
        self.button = Button(self.root, text="Submit", command=self.get_data, height=1, 
width=8, font="calibri 20 bold", background="green", foreground="white").grid(row=4, column=1, 
padx=100, pady=10)
```

Then, we define the main class called **Semantic_similarity_calculator** responsible for running the application and contain all the functions that perform comparison analysis and provides user interface.

The first window for user is created using the following standard tkinter widgets:

- **Label widget:** It is used to show text lines or images for the user, such as “Find similar products”, “Insert product id” and “insert W value” and to determine the format of the text as well as its position. User does not have the option however to interact with these texts
- **Entry widget:** It is used to display the box where the user can insert the inputs which are product id and weight of family ontology similarity index “W”

- **Button:** User can interact with this widget and can click on this button to perform a required action through command argument which calls the execution **get_data** function that we will explain later over the product number inserted by the user. It can also display text and images.

```
    def get_data(self):
        total_row = 0
        final_list = []
        filename = 'C:\\Users\\Hassan\\Desktop\\productslist.xlsx'
        product_name = self.entry.get()
        constant = self.entry1.get()
        comparison_iteration = self.compare_products(filename, product_name)
        if comparison_iteration is None:
            messagebox.showinfo(title="Alert", message="No product found,please check id
inserted")
        else:
            for iteration_i in comparison_iteration:
                final_list.append(
                    {
                        'product_name': iteration_i['product'],
                        'Similarity Index': float(constant) * (iteration_i['common_ic_fam.'] /
iteration_i['selected_prod._ic_fam']) + (1.0 - float(constant)) *
(iteration_i['common_ic_mat.'] / iteration_i['selected_prod._ic_mat'])
                    }
                )
            sort_list = sorted(final_list, key=itemgetter('Similarity Index'), reverse=True)

            screen = Tk()
            Label(screen, text="Most Similar Products", font=("Arial", 30)).grid(row=0,
columnspan=3)
            cols = ("Product Name", "Similarity Index")
            tree = ttk.Treeview(screen, column=cols, show='headings')
            for col in cols:
                tree.heading(col, text=col)
            tree.grid(row=1, column=2, columnspan=2)

            total_row = 10 if len(sort_list) > 10 else len(sort_list)

            for row in range(total_row):
                tree.insert("", "end", values=(sort_list[row]['product_name'],
sort_list[row]['Similarity Index']))

            screen.mainloop()
```

- **get_data ()** function is responsible mainly for displaying the final output to the user through different steps:

First it defines the file path and gets the product id inserted from the user through command argument. Then it calls **compare_products ()** function, which is responsible for comparing different products that will be discussed in details later, and passes product id and file path to it.

If product is not found, it will display this in a message to the user, otherwise it executes different iterations inside **compare_products()** function whose number is equal to the number of products in the excel file and it will get the required information related to the comparison analysis such as information content of selected product in both the ontologies as well as the common information content between the selected product and all others by appending them to the tuple that contain a dictionary where the key is the product id and value is the similarity index whose formula is not fully shown in the image but it is similar to the following despite using different syntax.

$$\text{Sim(Pa,Pb)=W}\times\left(\frac{\sum_{ci\in(Ga\cap Gb)} ICfamily\ ontology(ci)}{\sum_{cj\in(Ga)} ICfamily\ ontology(cj)}\right)+(1-W)\left(\frac{\sum_{ci\in(Ga\cap Gb)} ICfamily\ ontology(ci)}{\sum_{cj\in(Ga)} ICfamily\ ontology(cj)}\right)$$

Then the similar products are listed in descending order through **sorted** function where **itemgetter** is used to identify value of the key according to which sorting take place, which is Similarity index as shown in the image.

The last step in this function is to create another window to display in the interface and displayed after all iterations ends. This window is also created using tkinter **Treeview** widget that is used to display hierarchy of items for users where one or more attributes of each item can be displayed as columns and here it is used to display the result to the user in a tabular form that contain two columns, one contains product id and the other contains the corresponding Similarity index.

To ensure that only the most 10 similar products are displayed, the length of the sorted list is set to 10 if it is longer than 10 and it's kept as it is less.

```python
def compare_products(self, input_file, selected_product):
    selected_product_ic_family_ontology = 0
    selected_product_ic_material_ontology = 0
    product_list = []
    selected_family = ''
    selected_material = ''
    selected_base_material = ''
    selected_surf_finish = ''
    selected_surf_state = ''
    selected_complexity = ''
    sub_family_selected = []
    wb2 = load_workbook(filename=input_file)
    ws = wb2['Foglio3']
    for row in ws.iter_rows(min_row=2):
        try:
            if str(row[0].value) == selected_product and row[0].value is not None:
                selected_material = row[6].value
                selected_base_material = row[7].value
                selected_surf_finish = row[8].value
                selected_surf_state = row[9].value
                selected_family = row[11].value
                selected_complexity = int(row[17].value)
                if row[12].value is not '' and row[12].value not in sub_family_selected:
                    sub_family_selected.append(row[12].value)
                if row[13].value is not '' and row[13].value not in sub_family_selected:
                    sub_family_selected.append(row[13].value)
                if row[14].value is not '' and row[14].value not in sub_family_selected:
                    sub_family_selected.append(row[14].value)
                if row[15].value is not '' and row[15].value not in sub_family_selected:
                    sub_family_selected.append(row[15].value)
        except:
            continue
```

The rest of explanation is given to the comparison function in which we first define list of products and different variables that will be assigned to each product that are mentioned in previous slides.

Selected_product_ic_family_ontology () is IC of the selected product obtained from final product ontology by adding IC of different classes belonging to its graph. Same applies to **selected_product_ic_material_ontology** but IC is derived from concepts belonging to material ontology.

Then excel file is loaded through **load_workbook** function imported from openpyxl library and then the iteration over all the rows of the file is done by **ws.iter_rows** function also imported from the same library . At each iteration beginning, a check if the inserted product id value (row[0]) corresponds to the selected product by the user and if this condition is satisfied it assigns attributes to the selected product by importing each value from the excel file.

```
        if selected_material == 'A':
            selected_product_ic_material_ontology += material_a_ic
        elif selected_material == 'M':
            selected_product_ic_material_ontology += material_m_ic
        elif selected_material == 'G':
            selected_product_ic_material_ontology += material_g_ic
        elif selected_material == 'Z':
            selected_product_ic_material_ontology += material_z_ic
        elif selected_material == 'E':
            selected_product_ic_material_ontology += material_e_ic
        else:
            pass

        if selected_family == 'PN':
            selected_product_ic_family_ontology += pn_family_ic
        elif selected_family == 'AZ':
            selected_product_ic_family_ontology += az_family_ic
        elif selected_family == 'NS':
            selected_product_ic_family_ontology += ns_family_ic
        elif selected_family == 'ES':
            selected_product_ic_family_ontology += es_family_ic
        else:
            pass

        selected_product_ic_family_ontology += complexity_ic + sub_family_ic
        selected_product_ic_material_ontology += base_material_ic + surface_state_ic +
surface_finish_ic
```

After having assigned the attributes to the selected product, its IC in the two ontologies is calculated by adding the IC of each of its concepts, If statements are introduced here only for concepts that have distinctive information content such as product family and material type whose value differs between concepts, while for complexity, surface state, base material and surface finish the IC is same for all products and their IC is simply added without checking any condition. This is considered the first main step of the comparison function.

```
            for row in ws.iter_rows(min_row=2):
                common_ic_family_ontology = 0
                common_ic_material_ontology = 0
                sub_family_other_product = []
                try:
                    if str(row[0].value) != selected_product and str(row[0].value) is not
None:
                        if str(row[12].value) is not '' and str(row[12].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[12].value))
                        if str(row[13].value) is not '' and str(row[13].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[13].value))
                        if str(row[14].value) is not '' and str(row[14].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[14].value))
                        if str(row[15].value) is not '' and str(row[15].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[15].value))

                        if str(row[11].value) is selected_family and str(row[11].value) ==
'PN':
                            common_ic_family_ontology += pn_family_ic
                            for match in sub_family_selected:
                                if match in sub_family_other_product and str(row[0].value) not
in selected_product and match is not None:
                                    common_ic_family_ontology += sub_family_ic
                                    if str(selected_complexity) == str(row[17].value):
                                        common_ic_family_ontology += complexity_ic
                                    break
```

This is the second main loop, and it regards the selection of the attributes of the other products and then compares them with those of the selected product to calculate the common IC values.
Firstly, the subfamily of the other product is assigned by checking rows from 12 to 15 and assign the corresponding subfamily.
Later nested if statements to check equal attributes between products starting by product family passing by product subfamily and ending by complexity is performed to calculate the information content of common concepts. These statements are repeated for the remaining product families, ES, NS, AZ following the same logic.

```
                        if str(row[6].value) is selected_material and str(row[6].value) ==
'A':
                            common_ic_material_ontology += material_a_ic
                            if str(row[7].value) is selected_base_material:
                                common_ic_material_ontology += base_material_ic
                            if str(row[8].value) is selected_surf_finish:
                                common_ic_material_ontology += surface_finish_ic
                            if str(row[9].value) is selected_surf_state:
                                common_ic_material_ontology += surface_state_ic
                        elif str(row[6].value) is selected_material and str(row[6].value) ==
'M':
                            common_ic_material_ontology += material_a_ic
                            if str(row[7].value) is selected_base_material:
                                common_ic_material_ontology += base_material_ic
                            if str(row[8].value) is selected_surf_finish:
                                common_ic_material_ontology += surface_finish_ic
                            if str(row[9].value) is selected_surf_state:
                                common_ic_material_ontology += surface_state_ic
```

Here another set of if statements are introduced to calculate common IC coming from material ontology. It starts by comparing material types and it continues to check base material, surface finish and surface state and assign their IC only if products have same material type, otherwise common material IC is set to 0.
Same procedure is repeated for the five material types we have and their subclasses.

```
                product_list.append(
                    {
                        'product': row[0].value,
                        'common_ic_fam.': common_ic_family_ontology,
                        'selected_prod._ic_fam': selected_product_ic_family_ontology,
                        'common_ic_mat.': common_ic_material_ontology,
                        'selected_prod._ic_mat': selected_product_ic_material_ontology
                    }
                )
        except Exception as e:
            print(str(e))
            continue
```

At the end of comparison, and as a last step of this function the product is added to the tuple and put inside a dictionary together with common IC and IC of the selected product for both ontologies. This iteration is done for all products inside the file and stops once product id cell is found empty.

5.6.3 Running the application

1. **Adding new product:** The new product is added by inserting its attributes to the excel file.

2. **Inserting product id and weight of family ontology**

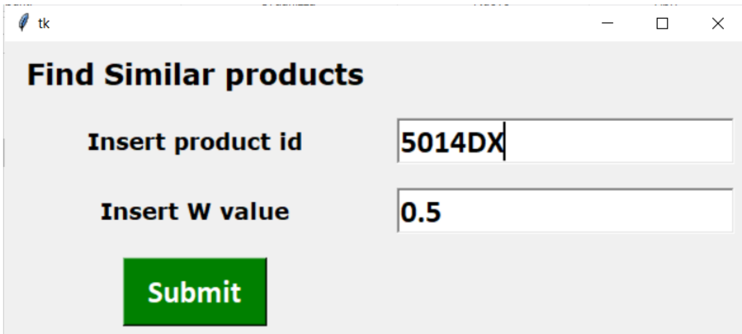


Figura 1 Screenshot of the similarity application: Datainput to look for similar products

3. **Error Message:**

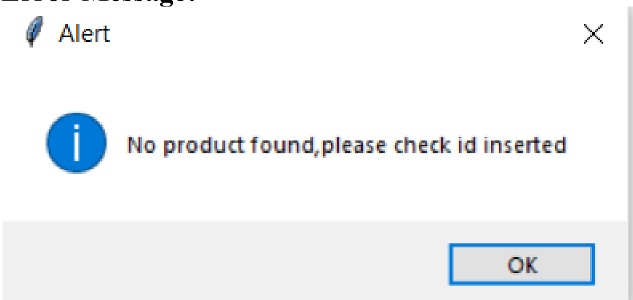


figure 2 Screenshot of the similarity application: Error Message

4. **Products list if product is found:**

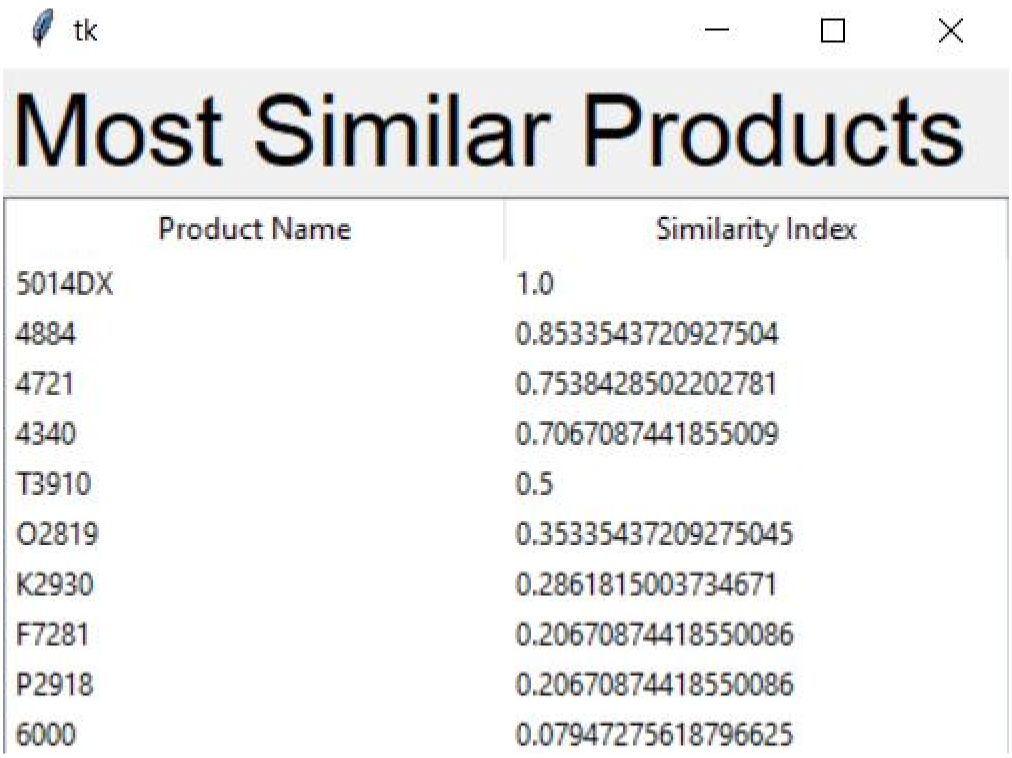


figure 3 Screenshot of the similarity application: Output indicating the most similar products

OBSERVATIONS AND CONCLUSION

Observations

By analysing the results, we can notice that the method is effective in analysing the common attributes between products such as final product and material ontologies and transferring them into meaningful numbers that indicate the level of similarity through a similarity index according to which they are sorted and displayed to the user. This becomes so useful when the designer needs to compare large number of products with many different attributes. In the first extreme case when the most similar product has a similarity index of 1, this means the presence of an identical product to the new order and the company can use previously defined production cycles for the new product without any modification and can avoid issues previously faced with the production of it. In the other extreme case where all similarity indices are zero, this indicates the absence of similar products and that the new product is completely different from those produced in the past, the designer will directly start defining everything from scratch instead of wasting his time in trying to find similar products by going through different documents related to that product. In the intermediate cases the designer can compare the new product to the most similar product instead of doing it for all products and can make necessary design and process modifications.

The biggest limitation of this method is the equal importance it gives to concepts with same number of descendants in the hierarchy despite the possible difference in their influence of determining the degree of similarity. Let us consider material ontology, both base material and surface finish have the same level of contribution to the similarity because they have no descendants, but this might not be true from the practical point of view and maybe our way is underestimating this difference in the contribution. The second limitation is related to the level of similarity, for example a product with complexity 3 is considered to have same similarity, which is zero, with products having complexity of either 1 or 2. This might not be again true from the practical point of view since a product of complexity 3 should be more similar to a product with complexity of 2 than it would be with a product of complexity of 1.

A solution could be to as ask expert designers and technicians to adjust the information content of each class according to their perception of the contribution it makes to the similarity.

Conclusion

The proposed similarity method covers the gap related to using categorical attributes in similarity assessment and allows to leverage on vaster product information compared to the common similarity methods that are mainly focused on numerical attributes. Moreover, the method is effective in automating information retrieval process which is known to be a time intensive process for engineers and designers during product development. Thus, it facilitates finding similar products in an immediate way and fulfills the requirements of accelerating knowledge retrieval process for OKP companies. However, numerical attributes still contain valuable product information, and this method can be more a complementary tool than a substitute to the numerical one. Therefore, a further improvement to the proposed method can be performed by combining both numerical and categorical similarities to estimate an overall similarity index between the products.

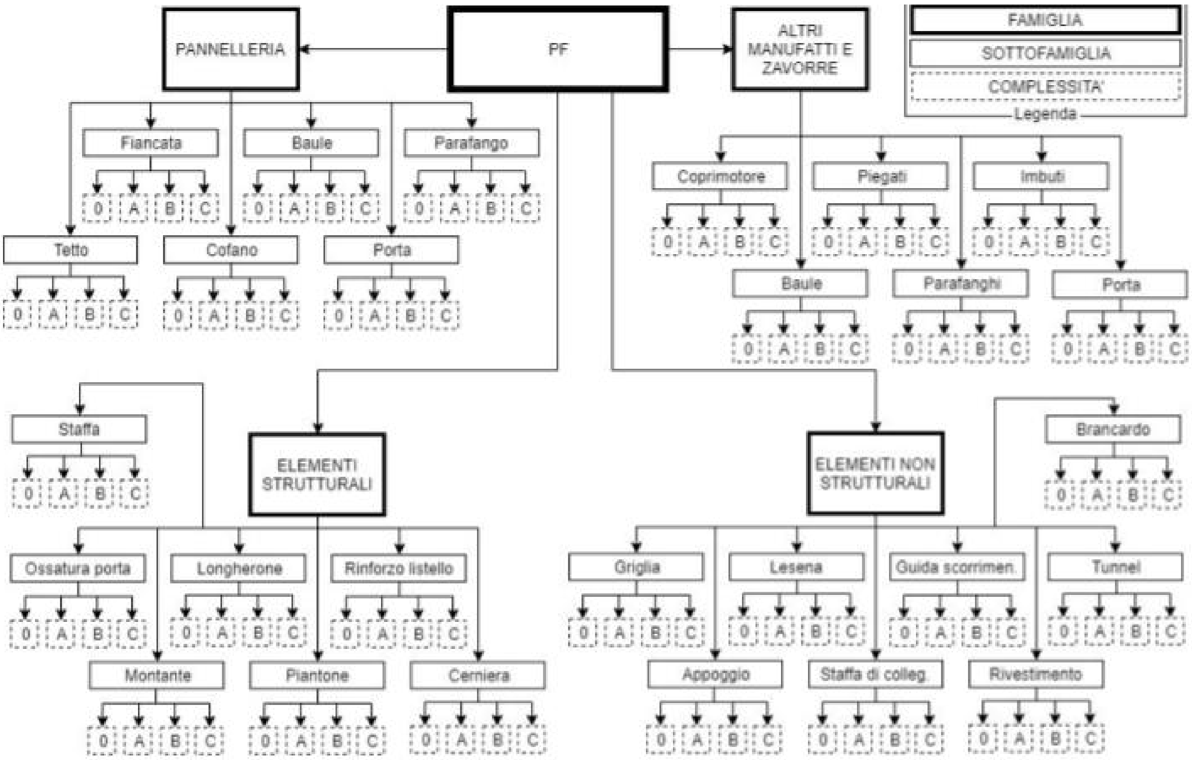
REFERENCES

1. Aamodt A, and Plaza E, (1994) "Case-based reasoning: foundational issues, methodological variations and system approaches''AI Communications, Vol. 7 No. 1, pp. 39-59.
2. Abdelhadi D, Noreddine G, and Abderrahim M, (2017), " An enhanced method to compute the similarity between concepts of ontology", Advances in Intelligent Systems and Computing 640, p4.
3. Adhikari A, Dutta B, Dutta A, Mondal D, Singh S, (2018) "An intrinsic information content-based semantic similarity measure considering the disjoint common subsumers of concepts of an ontology", Journal of the Association for Information Scence and Technology 69:1023-1034.
4. Balmisse, Meingan, & Passerini, (2007) "Technology Trends in Knowledge Management Tools" International Journal of Knowledge Management, vol. 3.
5. Ben Khedher A, Henry S, Bouras A (2011) "Integration between MES and Product Lifecycle Management", In: Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on, 1-8. (2011). doi: 10.1109/etfa.2011.6058993.
6. Bruno, G., Faveto, A., Traini, E., (2020) "An Open-Source Framework for The Storage and Reuse of Industrial knowledge through the Integration of PLM and MES", Management and Production Engineering Review,11, pp. 62–73.
7. Bruno, G. (2015), "Measuring product semantic similarity by exploiting a manufacturing process ontology", 6th IESM Conference, October 2015, Seville, Spain
8. Chakraborty, K., (1997) "Sustained competitive advantage: A resource-based framework", Advances in Competitiveness Research,5(1): p. 32-63
9. Clyde W. Holsapple,(2005) "The inseparability of modern knowledge management and computer-based technology" Journal of Knowledge Management
10. D'Antonio G, Macheda L, Bedolla J S, Chiabert P, "PLM-MES Integration to Support Industry 4.0", 14th IFIP International Conference on Product Lifecycle Management (PLM), Jul 2017, Seville, Spain. pp.129-137, ff10.1007/978-3-319-72905-3_12ff. ffhal-01764194
11. Hong G, Xue D, Tu Y, (2010) "Rapid identification of the optimal product configuration and its parameters based on customer-centric product modeling for one-ofa-kind production" Computers in Industry, 61(3), 270–279
12. Jokinen, L. Leino, S., (2019) "Hidden product knowledge, problems and solutions" ,29th International Conference on Flexible Automation and Intelligent Manufacturing, Limerick, Ireland.
13. Kharbat F, El-Ghalayin H, (2008) "Building ontology from knowledge base systems" Data Mining in Medical and Biological Research. I-Tech, Vienna, Austria, 320p.
14. Kyoung-Yun Kim, David G. Manley, Hyungjeong Yang, (2006) "Ontology-based assembly design and information sharing for collaborativeproduct development", Computer-Aided Design 38,1233–1250
15. Li, Z. & Liu, M (2004) "Review of product information retrieval: representation and indexing". In: Proceedings of ASME 2004 Computers and Information in Engineering Conference, 4: 971-979, Salt Lake
16. Meng L, Gu J, Zhou Z (2012) "A new model of information content based on concept's topology for measuring semantic similarity in WordNet", International Journal of Grid and Distributed Computing 5:81-93.
17. Muhammad B. Qureshi (2009) "Analysis of Knowledge Management Tools"
18. Pirró, G. (2009) "A semantic similarity metric combining features and intrinsic information content" Data& Knowledge Engineering, 68, 1289-1308.
19. Poeschla S, Liebb J, Wirth F, Bauernhanslc T, (2017)"Expert systems in special machinery: Increasing the productivity of processes in commissioning", The 50th CIRP Conference on Manufacturing Systems, p546.
20. Rada, R., Mili, H., Bicknell, E., and Blettner, M. (1989) "Development and Application of a Metric on Semantic Nets", IEEE Transactions on Systems, Man, and Cybernetics, 19(1):17-30.
21. R. Baeza-Yates, B. Ribeiro-Neto., (1999) "Modern Information Retrieval". ACM Press; Addison-Wesley: New York; Harlow, England; Reading, Mass.
22. Resnik O.(1999) "Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity and Natural Language" Journal of Artificial Intelligence Research, 11, 95-130.

23. Ryutarou Ohbuchi, Tsuyoshi Takei, (2003), “Shape similarity comparison of 3D models using alpha shapes” In the proceedings of the Pacific Graphics, Canmore, Canada, p2
24. Sánchez, D., Isern, D., Batet, M., Valls, A.,(2012) “Ontology-based semantic similarity: A new feature-based approach”, Expert Systems with Applications.
25. Sánchez D, Batet M (2011) Semantic similarity estimation in the biomedical domain: an ontology-based information-theoretic perspective Journal of Biomedical Informatics 44:749-759
26. Seco, N., Veale, T., & Hayes, J. (2004) “An Intrinsic Information Content Metric for Semantic Similarity in WordNet” In R. López de Mántaras & L. Saitta (Eds.), 16th Eureopean Conference on ArtificialIntelligence, ECAI.
27. Siemens 2018, “Mass customization: The factory of the future”, <https://new.siemens.com/global/en/company/stories/industry/the-factory-of-the-future.html>
28. Slimani, T. Ben Yaghlane, B. and Mellouli, K. (2006) “A New Similarity Measure based on Edge Counting”, World Academy of Science, Engineering and Technology, PP 34-38.
29. Stark J. “Product Life Cycle Management” 21sr century Paradigm for product Realization, vol1, third edition, p8
30. Tahir A J, Mutahar Sa, Imgyu K, and Soonhung H. (2020) “Web-based Product Data Visualization and Feedback between PLM and MES”, In The 25th International Conference on 3D Web Technology (Web3D ’20), November 9–13, 2020, Virtual Event, Republic of Korea. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3424616.3424694
31. Terzi S, Bouras D, Dutta M, Garetti D, (2010) “Product lifecycle management from its history to its new role” Int. J. Prod. Lifec. Manag. 4 (4) 360–389.
32. Traini E, Bruno G, Awouda A, Chiabert P, Lombardi F, (2019)“Integration Between PLM and MES for One-of-a-Kind Production”
33. Tu Y, Dean P, (2010)“One-of-a-Kind-Production”, p12
34. Tu YL, Chu XL, Yang WY, (2000) “Computer aided process planning in agile one-of-a-kind production”. Int J Comput Ind 41:99–110
35. Tversky, A. (1977)”Features of Similarity”, Psycological Review.
36. Umble E J , Haft R R, Umble M M,(2003) “Enterprise resource planning: Implementation procedures and critical success factors” European Journal of Operational Research, 146(2), 241-257 (2003). doi: 10.1016/S0377-2217(02)00547-
37. Wortmann J C, (1991) “Towards One-Of-A-Kind Production: The Future of European Industry” in Eloranta, E (ed) Advances in Production Management Systems. Elsevier
38. Yuan Q, Yu Z, Wang K, (2013)“A new model of information content for measuring the semantic similarity between concept” ,Proceedings of the Second International Conference on Cloud Computing and Big Data.

APPENDICES

Appendix 1: Original classification of final products of Eurodies Company



Appendix 2: Information content for material classes

Material Type	IC	Base Material	IC	Surface Finish (For Each material Type)	IC	Surface state (For each material Type)	IC
A	0,4094	AL5	1	STD	1	E(MC)	1
		AL6	1	STD-NP	1	U(MB)	1
		ALX	1	STD-TZ	1	B	1
		TL090	1	OUT	1	C	1
		TL091	1	OUT-NP	1		
		TL094	1	OUT-TZ	1		
		TL100	1	HSA-NP	1		
		TL114	1	HSA-TZ	1		
		AC300	1	HYF	1		
		TL1550-220	1	WWP	1		
		TL127	1	UFO-TZ	1		
		AA6111	1	H111	1		
		ALMG3	1	IIC-TZ	1		
		AA6000	1	HDI-TZ	1		
		5754	1	T4	1		
		6016T4EDT	1	UFO-T4	1		
		EN-AW5083	1	DL	1		
		EN-AW6014	1	UFO	1		
G	0.2388	DX51D	1	UNCOATED	1	O	1
		DX52D	1			P-O	1
		DX53D(CR2)	1			E(MC)	1
		DX54D(CR3)	1			U(MB)	1
		DX55D	1				
		DX56D(CR4)	1				
		DX57D(CR5)	1				
		CR1(DX52D)	1				
		CR2(DX53D)	1				
		CR3(DX54D)	1				
		CR4(DX55D)	1				
		CR5(DX56D)	1				
		CR3-NON_IF	1				
		HX250	1				
		HX160YD(CR160IF)	1				
		HX180	1				
		HX180YD(CR180IF)	1				
		HX180BD(CR180BH)	1				
		HX220YD(CR210IF)	1				

		HX220BD(CR210BD)	1				
		HX220PD	1				
		HX260YD(CR240IF)	1				
		HX260BD(CR240BH)	1				
		HX260LAD(CR240LA)	1				
		HX300YD	1				
		HX300BD	1				
		HX300LAD(CR270LA)	1				
		HX340BD	1				
		HX340LAD(CR300LA)	1				
		HX380LAD(CR340LA)	1				
		HX420LAD(CR380LA)	1				
		HX460LAD(CR420LA)	1				
		HX500LAD((CR460LA))	1				
		HX180LAD	1				
		CR210LA(ZSTE220Z)	1				
		CR240LA(HX260LAD)	1				
		CR270LA(HX300LAD)	1				
		CR300LA(HX340LAD)	1				
		CR340LA(HX380LAD)	1				
		CR380LA(HX420LAD)	1				
		CR420LA(HX460LAD)	1				
		CR160IF(HX160YD)	1				
		CR180IF(HX180YD)	1				
		CR210IF(HX220YD)	1				
		CR240IF(HX260YD)	1				
		CR180BH(HX180BD)	1				
		CR210BH(HX220BD)	1				
		CR240BH(HX260BD)	1				
		CR210	1				
		CR290Y490T-DP(HC300XD/HCT500X/DP500)	1				
		CR330Y590T-DP(HC340XD/HCT600X/DP600)	1				
		CR420Y780T-DP	1				
		CR440Y780T-DP(HC450XD/HCT780X/DP780)	1				
		CR550Y980T-DP	1				
		CR590Y980T-DP(HC600XD/HCT980X/DP980)	1				
		CR700Y980T-DP(HC660XD/DP980HY)	1				
		CR400Y690T-TR(HC410TD/HCT690T/TRIP700)	1				
		CR450Y780T-TR(HC470TD/HCT780T/TRIP800)	1				

		CR570Y780T-CP(HC600CD/HCT780C/CP800)	1				
		CR780Y980T-CP(HC800CD/HCT980C/CP980)	1				
		CR700Y980T	1				
		CR700Y980T-MP-LCE	1				
		CR340Y590T-DP	1				
		CR700Y980T-MP	1				
		HR300Y450T-FB(HDT450F/FB450)	1				
		HR440Y580T-FB(HDT560F/FB590)	1				
		HR600Y780T-FB(FB780)	1				
		HR330Y580T-DP(HDT580X/DP600)	1				
		HR660Y760T-CP(HD680CD/HDT780C/CP800)	1				
		HR900Y1180T-MS(HDT1200M/MS1200)	1				
		HR550	1				
		DC01	1				
		DC04	1				
		DC04A	1				
		DC05	1				
		DC06	1				
		DD11	1				
		DD12	1				
		DD14	1				
		H260PD	1				
		H280LA	1				
		H340LAD	1				
		H420LA	1				
		HC180BD	1				
		HC180BH	1				
		HC180YD	1				
		HC220Y	1				
		HC300XD	1				
		HC340X	1				
		HC340XD	1				
		HC450XD	1				
		HC660XD	1				
		CR440	1				
		BH260	1				
		FE340	1				
		FEDD12	1				
		HXT600XD	1				
		P04	1				

		HX220LAD	1				
		HX380	1				
		S355 MC 10149-2 MC	1				
		LAC420Y480T	1				
		LAH420Y480T	1				
		CR330Y590T-DH	1				
		CR380LA-TRB	1				
		DP1000	1				
		DP800	1				
		FE420	1				
		CR260LA	1				
		HR340LA	1				
		DPC340Y590T	1				
		FEP04	1				
		DC02	1				
		CR980T/700Y	1				
Z	0.227	DX51D	1	GI40/40	1	O	1
		DX52D	1	GI50/50	1	P-O	1
		DX53D(CR2)	1	GI60/60	1	E(MC)	1
		DX54D(CR3)	1	GI70/70	1	U(MB)	1
		DX55D	1	GI150/150	1		
		DX56D(CR4)	1	Z100	1		
		DX57D(CR5)	1	GI20/20	1		
		CR1(DX52D)	1	Z140	1		
		CR2(DX53D)	1	HD60G60G	1		
		CR3(DX54D)	1	FFA-TZ	1		
		CR4(DX55D)	1				
		CR5(DX56D)	1				
		CR3-NON_IF	1				
		HX250	1				
		HX160YD(CR160IF)	1				
		HX180	1				
		HX180YD(CR180IF)	1				
		HX180BD(CR180BH)	1				
		HX220YD(CR210IF)	1				
		HX220BD(CR210BD)	1				
		HX220PD	1				
		HX260YD(CR240IF)	1				
		HX260BD(CR240BH)	1				
		HX260LAD(CR240LA)	1				
		HX300YD	1				
		HX300BD	1				
		HX300LAD(CR270LA)	1				
		HX340BD	1				
		HX340LAD(CR300LA)	1				
		HX380LAD(CR340LA)	1				

		HX420LAD(CR380LA)	1				
		HX460LAD(CR420LA)	1				
		HX500LAD((CR460LA))	1				
		HX180LAD	1				
		CR210LA(ZSTE220Z)	1				
		CR240LA(HX260LAD)	1				
		CR270LA(HX300LAD)	1				
		CR300LA(HX340LAD)	1				
		CR340LA(HX380LAD)	1				
		CR380LA(HX420LAD)	1				
		CR420LA(HX460LAD)	1				
		CR160IF(HX160YD)	1				
		CR180IF(HX180YD)	1				
		CR210IF(HX220YD)	1				
		CR240IF(HX260YD)	1				
		CR180BH(HX180BD)	1				
		CR210BH(HX220BD)	1				
		CR240BH(HX260BD)	1				
		CR210	1				
		CR290Y490T-DP(HC300XD/HCT500X/DP500)	1				
		CR330Y590T-DP(HC340XD/HCT600X/DP600)	1				
		CR420Y780T-DP	1				
		CR440Y780T-DP(HC450XD/HCT780X/DP780)	1				
		CR550Y980T-DP	1				
		CR590Y980T-DP(HC600XD/HCT980X/DP980)	1				
		CR700Y980T-DP(HC660XD/DP980HY)	1				
		CR400Y690T-TR(HC410TD/HCT690T/TRIP700)	1				
		CR450Y780T-TR(HC470TD/HCT780T/TRIP800)	1				
		CR570Y780T-CP(HC600CD/HCT780C/CP800)	1				
		CR780Y980T-CP(HC800CD/HCT980C/CP980)	1				
		CR700Y980T	1				
		CR700Y980T-MP-LCE	1				
		CR340Y590T-DP	1				
		CR700Y980T-MP	1				
		HR300Y450T-FB(HDT450F/FB450)	1				
		HR440Y580T-FB(HDT560F/FB590)	1				

		HR600Y780T-FB(FB780)	1				
		HR330Y580T-DP(HDT580X/DP600)	1				
		HR660Y760T-CP(HD680CD/HDT780C/CP800)	1				
		HR900Y1180T-MS(HDT1200M/MS1200)	1				
		HR550	1				
		DC01	1				
		DC04	1				
		DC04A	1				
		DC05	1				
		DC06	1				
		DD11	1				
		DD12	1				
		DD14	1				
		H260PD	1				
		H280LA	1				
		H340LAD	1				
		H420LA	1				
		HC180BD	1				
		HC180BH	1				
		HC180YD	1				
		HC220Y	1				
		HC300XD	1				
		HC340X	1				
		HC340XD	1				
		HC450XD	1				
		HC660XD	1				
		CR440	1				
		BH260	1				
		FE340	1				
		FEDD12	1				
		HXT600XD	1				
		P04	1				
		HX220LAD	1				
		HX380	1				
		S355 MC 10149-2 MC	1				
		LAC420Y480T	1				
		LAH420Y480T	1				
		CR330Y590T-DH	1				
		CR380LA-TRB	1				
		DP1000	1				
		DP800	1				
		FE420	1				
		CR260LA	1				

		HR340LA	1				
		DPC340Y590T	1				
		FEP04	1				
		DC02	1				
		CR980T/700Y	1				
E	0.231	DX51D	1	EG29/29	1	O	1
		DX52D	1	EG50/50	1	P-O	1
		DX53D(CR2)	1	EG53/53	1	E(MC)	1
		DX54D(CR3)	1	EG47/47	1	U(MB)	1
		DX55D	1	ZE40/40	1		
		DX56D(CR4)	1	ZE50/50	1		
		DX57D(CR5)	1	ZE75/75	1		
		CR1(DX52D)	1				
		CR2(DX53D)	1				
		CR3(DX54D)	1				
		CR4(DX55D)	1				
		CR5(DX56D)	1				
		CR3-NON_IF	1				
		HX250	1				
		HX160YD(CR160IF)	1				
		HX180	1				
		HX180YD(CR180IF)	1				
		HX180BD(CR180BH)	1				
		HX220YD(CR210IF)	1				
		HX220BD(CR210BD)	1				
		HX220PD	1				
		HX260YD(CR240IF)	1				
		HX260BD(CR240BH)	1				
		HX260LAD(CR240LA)	1				
		HX300YD	1				
		HX300BD	1				
		HX300LAD(CR270LA)	1				
		HX340BD	1				
		HX340LAD(CR300LA)	1				
		HX380LAD(CR340LA)	1				
		HX420LAD(CR380LA)	1				
		HX460LAD(CR420LA)	1				
		HX500LAD((CR460LA))	1				
		HX180LAD	1				
		CR210LA(ZSTE220Z)	1				
		CR240LA(HX260LAD)	1				
		CR270LA(HX300LAD)	1				
		CR300LA(HX340LAD)	1				
		CR340LA(HX380LAD)	1				
		CR380LA(HX420LAD)	1				
		CR420LA(HX460LAD)	1				
		CR160IF(HX160YD)	1				

		CR180IF(HX180YD)	1				
		CR210IF(HX220YD)	1				
		CR240IF(HX260YD)	1				
		CR180BH(HX180BD)	1				
		CR210BH(HX220BD)	1				
		CR240BH(HX260BD)	1				
		CR210	1				
		CR290Y490T-DP(HC300XD/HCT500X/DP500)	1				
		CR330Y590T-DP(HC340XD/HCT600X/DP600)	1				
		CR420Y780T-DP	1				
		CR440Y780T-DP(HC450XD/HCT780X/DP780)	1				
		CR550Y980T-DP	1				
		CR590Y980T-DP(HC600XD/HCT980X/DP980)	1				
		CR700Y980T-DP(HC660XD/DP980HY)	1				
		CR400Y690T-TR(HC410TD/HCT690T/TRIP700)	1				
		CR450Y780T-TR(HC470TD/HCT780T/TRIP800)	1				
		CR570Y780T-CP(HC600CD/HCT780C/CP800)	1				
		CR780Y980T-CP(HC800CD/HCT980C/CP980)	1				
		CR700Y980T	1				
		CR700Y980T-MP-LCE	1				
		CR340Y590T-DP	1				
		CR700Y980T-MP	1				
		HR300Y450T-FB(HDT450F/FB450)	1				
		HR440Y580T-FB(HDT560F/FB590)	1				
		HR600Y780T-FB(FB780)	1				
		HR330Y580T-DP(HDT580X/DP600)	1				
		HR660Y760T-CP(HD680CD/HDT780C/CP800)	1				
		HR900Y1180T-MS(HDT1200M/MS1200)	1				
		HR550	1				
		DC01	1				
		DC04	1				
		DC04A	1				
		DC05	1				
		DC06	1				
		DD11	1				

		DD12	1				
		DD14	1				
		H260PD	1				
		H280LA	1				
		H340LAD	1				
		H420LA	1				
		HC180BD	1				
		HC180BH	1				
		HC180YD	1				
		HC220Y	1				
		HC300XD	1				
		HC340X	1				
		HC340XD	1				
		HC450XD	1				
		HC660XD	1				
		CR440	1				
		BH260	1				
		FE340	1				
		FEDD12	1				
		HXT600XD	1				
		P04	1				
		HX220LAD	1				
		HX380	1				
		S355 MC 10149-2 MC	1				
		LAC420Y480T	1				
		LAH420Y480T	1				
		CR330Y590T-DH	1				
		CR380LA-TRB	1				
		DP1000	1				
		DP800	1				
		FE420	1				
		CR260LA	1				
		HR340LA	1				
		DPC340Y590T	1				
		FEP04	1				
		DC02	1				
		CR980T/700Y	1				
M	0.236	DX51D	1	ZM35/35	1	O	1
		DX52D	1	ZM40/40	1	P-O	1
		DX53D(CR2)	1	ZM150/150	1	E(MC)	1
		DX54D(CR3)	1			U(MB)	1
		DX55D	1				
		DX56D(CR4)	1				
		DX57D(CR5)	1				
		CR1(DX52D)	1				
		CR2(DX53D)	1				
		CR3(DX54D)	1				

		CR4(DX55D)	1				
		CR5(DX56D)	1				
		CR3-NON_IF	1				
		HX250	1				
		HX160YD(CR160IF)	1				
		HX180	1				
		HX180YD(CR180IF)	1				
		HX180BD(CR180BH)	1				
		HX220YD(CR210IF)	1				
		HX220BD(CR210BD)	1				
		HX220PD	1				
		HX260YD(CR240IF)	1				
		HX260BD(CR240BH)	1				
		HX260LAD(CR240LA)	1				
		HX300YD	1				
		HX300BD	1				
		HX300LAD(CR270LA)	1				
		HX340BD	1				
		HX340LAD(CR300LA)	1				
		HX380LAD(CR340LA)	1				
		HX420LAD(CR380LA)	1				
		HX460LAD(CR420LA)	1				
		HX500LAD((CR460LA))	1				
		HX180LAD	1				
		CR210LA(ZSTE220Z)	1				
		CR240LA(HX260LAD)	1				
		CR270LA(HX300LAD)	1				
		CR300LA(HX340LAD)	1				
		CR340LA(HX380LAD)	1				
		CR380LA(HX420LAD)	1				
		CR420LA(HX460LAD)	1				
		CR160IF(HX160YD)	1				
		CR180IF(HX180YD)	1				
		CR210IF(HX220YD)	1				
		CR240IF(HX260YD)	1				
		CR180BH(HX180BD)	1				
		CR210BH(HX220BD)	1				
		CR240BH(HX260BD)	1				
		CR210	1				
		CR290Y490T-DP(HC300XD/HCT500X/DP500)	1				
		CR330Y590T-DP(HC340XD/HCT600X/DP600)	1				
		CR420Y780T-DP	1				
		CR440Y780T-DP(HC450XD/HCT780X/DP780)	1				

		CR550Y980T-DP	1				
		CR590Y980T-DP(HC600XD/HCT980X/DP980)	1				
		CR700Y980T-DP(HC660XD/DP980HY)	1				
		CR400Y690T-TR(HC410TD/HCT690T/TRIP700)	1				
		CR450Y780T-TR(HC470TD/HCT780T/TRIP800)	1				
		CR570Y780T-CP(HC600CD/HCT780C/CP800)	1				
		CR780Y980T-CP(HC800CD/HCT980C/CP980)	1				
		CR700Y980T	1				
		CR700Y980T-MP-LCE	1				
		CR340Y590T-DP	1				
		CR700Y980T-MP	1				
		HR300Y450T-FB(HDT450F/FB450)	1				
		HR440Y580T-FB(HDT560F/FB590)	1				
		HR600Y780T-FB(FB780)	1				
		HR330Y580T-DP(HDT580X/DP600)	1				
		HR660Y760T-CP(HD680CD/HDT780C/CP800)	1				
		HR900Y1180T-MS(HDT1200M/MS1200)	1				
		HR550	1				
		DC01	1				
		DC04	1				
		DC04A	1				
		DC05	1				
		DC06	1				
		DD11	1				
		DD12	1				
		DD14	1				
		H260PD	1				
		H280LA	1				
		H340LAD	1				
		H420LA	1				
		HC180BD	1				
		HC180BH	1				
		HC180YD	1				
		HC220Y	1				
		HC300XD	1				
		HC340X	1				

		HC340XD	1				
		HC450XD	1				
		HC660XD	1				
		CR440	1				
		BH260	1				
		FE340	1				
		FEDD12	1				
		HXT600XD	1				
		P04	1				
		HX220LAD	1				
		HX380	1				
		S355 MC 10149-2 MC	1				
		LAC420Y480T	1				
		LAH420Y480T	1				
		CR330Y590T-DH	1				
		CR380LA-TRB	1				
		DP1000	1				
		DP800	1				
		FE420	1				
		CR260LA	1				
		HR340LA	1				
		DPC340Y590T	1				
		FEP04	1				
		DC02	1				
		CR980T/700Y	1				

Appendix 3: Full Python Code

```
from operator import itemgetter
from tkinter import StringVar, Tk, Label, Entry, Button, ttk, messagebox
from openpyxl import load_workbook
#Information content for Family ontology
sub_family_ic = 0.708369
complexity_ic = 1
pn_family_ic = 0.322853739
es_family_ic = 0.29163102
az_family_ic = 0.322853739
ns_family_ic =0.264449096

#Information Content for Material Ontology
material_a_ic = 0.40958
material_m_ic = 0.23621
material_g_ic = 0.238838
material_z_ic = 0.22734
material_e_ic = 0.231081
base_material_ic = 1
surface_finish_ic = 1
surface_state_ic = 1


class Semantic_simlarity_calculator:

    def __init__(self):
        self.root = Tk()
        self.textEntry = StringVar()
        self.label = Label(self.root, text="Find Similar products", font="Verdana 18
bold").grid(row=1, column=1, padx=10,pady=10)
        self.label1 = Label(self.root, text="Insert product id ", font="Verdana 14
bold").grid(row=2, column=1, padx=10, pady=10)
        self.entry = Entry(self.root, bd=2, width=20, font="calibri 20 bold")
        self.label2 = Label(self.root, text="Insert W value", font="Verdana 14 bold").grid(row=3,
column=1, padx=10, pady=10)
        self.entry1 = Entry(self.root, bd=2, width=20, font="calibri 20 bold")
        self.entry.grid(row=2, column=2, padx=10, pady=10)
        self.entry1.grid(row=3, column=2, padx=10, pady=10)
        self.button = Button(self.root, text="Submit", command=self.get_data, height=1, width=8,
font="calibri 20 bold", background="green", foreground="white").grid(row=4, column=1, padx=100,
pady=10)

    def get_data(self):
        total_row = 0
        final_list = []
        filename = 'C:\\Users\\Hassan\\Desktop\\productslist.xlsx'
        product_name = self.entry.get()
        constant = self.entry1.get()
        comparison_iteration = self.compare_products(filename, product_name)
        if comparison_iteration is None:
            messagebox.showinfo(title="Alert", message="No product found,please check id
inserted")
        else:
            for iteration_i in comparison_iteration:
                final_list.append(
                    {
                        'product_name': iteration_i['product'],
```

```
                        'Similarity Index': float(constant) * (iteration['common_ic_fam.'] /
iteration_i['selected_prod._ic_fam']) + (1.0 - float(constant)) * (iteration_i['common_ic_mat.'] /
iteration_i['selected_prod._ic_mat'])
                    }
                )
            sort_list = sorted(final_list, key=itemgetter('Similarity Index'), reverse=True)

            screen = Tk()
            Label(screen, text="Most Similar Products", font=("Arial", 30)).grid(row=0,
columnspan=3)
            cols = ("Product Name", "Similarity Index")
            tree = ttk.Treeview(screen, column=cols, show='headings')
            for col in cols:
                tree.heading(col, text=col)
            tree.grid(row=1, column=2, columnspan=2)

            total_row = 10 if len(sort_list) > 10 else len(sort_list)

            for row in range(total_row):
                tree.insert("", "end", values=(sort_list[row]['product_name'],
sort_list[row]['Similarity Index']))

            screen.mainloop()


    def compare_products(self, input_file, selected_product):
        selected_product_ic_family_ontology = 0
        selected_product_ic_material_ontology = 0
        product_list = []
        selected_family = ''
        selected_material = ''
        selected_base_material = ''
        selected_surf_finish = ''
        selected_surf_state = ''
        selected_complexity = ''
        sub_family_selected = []
        wb2 = load_workbook(filename=input_file)
        ws = wb2['Foglio3']
        for row in ws.iter_rows(min_row=2):
            try:
                if str(row[0].value) == selected_product and row[0].value is not None:
                    selected_material = row[6].value
                    selected_base_material = row[7].value
                    selected_surf_finish = row[8].value
                    selected_surf_state = row[9].value
                    selected_family = row[11].value
                    selected_complexity = int(row[17].value)
                    if row[12].value is not '' and row[12].value not in sub_family_selected:
                        sub_family_selected.append(row[12].value)
                    if row[13].value is not '' and row[13].value not in sub_family_selected:
                        sub_family_selected.append(row[13].value)
                    if row[14].value is not '' and row[14].value not in sub_family_selected:
                        sub_family_selected.append(row[14].value)
                    if row[15].value is not '' and row[15].value not in sub_family_selected:
                        sub_family_selected.append(row[15].value)
            except:
                continue
```

```
        if selected_material == 'A':
            selected_product_ic_material_ontology += material_a_ic
        elif selected_material == 'M':
            selected_product_ic_material_ontology += material_m_ic
        elif selected_material == 'G':
            selected_product_ic_material_ontology += material_g_ic
        elif selected_material == 'Z':
            selected_product_ic_material_ontology += material_z_ic
        elif selected_material == 'E':
            selected_product_ic_material_ontology += material_e_ic
        else:
            pass

        if selected_family == 'PN':
            selected_product_ic_family_ontology += pn_family_ic
        elif selected_family == 'AZ':
            selected_product_ic_family_ontology += az_family_ic
        elif selected_family == 'NS':
            selected_product_ic_family_ontology += ns_family_ic
        elif selected_family == 'ES':
            selected_product_ic_family_ontology += es_family_ic
        else:
            pass

        selected_product_ic_family_ontology += complexity_ic + sub_family_ic
        selected_product_ic_material_ontology += base_material_ic + surface_state_ic +
surface_finish_ic

        if selected_family == '':
            return
        else:
            for row in ws.iter_rows(min_row=2):
                common_ic_family_ontology = 0
                common_ic_material_ontology = 0
                sub_family_other_product = []
                try:
                    if str(row[0].value) != selected_product and str(row[0].value) is not None:
                        if str(row[12].value) is not '' and str(row[12].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[12].value))
                        if str(row[13].value) is not '' and str(row[13].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[13].value))
                        if str(row[14].value) is not '' and str(row[14].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[14].value))
                        if str(row[15].value) is not '' and str(row[15].value) not in
sub_family_other_product:
                            sub_family_other_product.append(str(row[15].value))

                        if str(row[11].value) is selected_family and str(row[11].value) == 'PN':
                            common_ic_family_ontology += pn_family_ic
                            for match in sub_family_selected:
                                if match in sub_family_other_product and str(row[0].value) not in
selected_product and match is not None:
                                    common_ic_family_ontology += sub_family_ic
                                    if str(selected_complexity) == str(row[17].value):
                                        common_ic_family_ontology += complexity_ic
```

```
                        break
                elif str(row[11].value) is selected_family and str(row[11].value) == 'AZ':
                    common_ic_family_ontology += pn_family_ic
                    for match in sub_family_selected:
                        if match in sub_family_other_product and str(row[0].value) not in
selected_product and match is not None:
                            common_ic_family_ontology += sub_family_other_ic
                            if str(selected_complexity) == str(row[17].value):
                                common_ic_family_ontology += complexity_ic
                            break
                elif str(row[11].value) is selected_family and str(row[11].value) == 'NS':
                    common_ic_family_ontology += pn_family_ic
                    for match in sub_family_selected:
                        if match in sub_family_other_product and str(row[0].value) not in
selected_product and match is not None:
                            common_ic_family_ontology += sub_family_ic
                            if str(selected_complexity) == str(row[17].value):
                                common_ic_family_ontology += complexity_ic
                            break
                elif str(row[11].value) is selected_family and str(row[11].value) == 'ES':
                    common_ic_family_ontology += pn_family_ic
                    for match in sub_family_selected:
                        if match in sub_family_other_product and str(row[0].value) not in
selected_product and match is not None:
                            common_ic_family_ontology += sub_family_other_ic
                            if str(selected_complexity) == str(row[17].value):
                                common_ic_family_ontology += complexity_ic
                            break
                else:
                    pass

                if str(row[6].value) is selected_material and str(row[6].value) == 'A':
                    common_ic_material_ontology += material_a_ic
                    if str(row[7].value) is selected_base_material:
                        common_ic_material_ontology += base_material_ic
                    if str(row[8].value) is selected_surf_finish:
                        common_ic_material_ontology += surface_finish_ic
                    if str(row[9].value) is selected_surf_state:
                        common_ic_material_ontology += surface_state_ic
                elif str(row[6].value) is selected_material and str(row[6].value) == 'M':
                    common_ic_material_ontology += material_a_ic
                    if str(row[7].value) is selected_base_material:
                        common_ic_material_ontology += base_material_ic
                    if str(row[8].value) is selected_surf_finish:
                        common_ic_material_ontology += surface_finish_ic
                    if str(row[9].value) is selected_surf_state:
                        common_ic_material_ontology += surface_state_ic
                elif str(row[6].value) is selected_material and str(row[6].value) == 'G':
                    common_ic_material_ontology += material_a_ic
                    if str(row[7].value) is selected_base_material:
                        common_ic_material_ontology += base_material_ic
                    if str(row[8].value) is selected_surf_finish:
                        common_ic_material_ontology += surface_finish_ic
                    if str(row[9].value) is selected_surf_state:
                        common_ic_material_ontology += surface_state_ic
                elif str(row[6].value) is selected_material and str(row[6].value) == 'Z':
                    common_ic_material_ontology += material_a_ic
                    if str(row[7].value) is selected_base_material:
```

```
                                common_ic_material_ontology += base_material_ic
                            if str(row[8].value) is selected_surf_finish:
                                common_ic_material_ontology += surface_finish_ic
                            if str(row[9].value) is selected_surf_state:
                                common_ic_material_ontology += surface_state_ic
                        elif str(row[6].value) is selected_material and str(row[6].value) == 'E':
                            common_ic_material_ontology += material_a_ic
                            if str(row[7].value) is selected_base_material:
                                common_ic_material_ontology += base_material_ic
                            if str(row[8].value) is selected_surf_finish:
                                common_ic_material_ontology += surface_finish_ic
                            if str(row[9].value) is selected_surf_state:
                                common_ic_material_ontology += surface_state_ic
                        else:
                            pass

                        product_list.append(
                            {
                                'product': row[0].value,
                                'common_ic_fam.': common_ic_family_ontology,
                                'selected_prod._ic_fam': selected_product_ic_family_ontology,
                                'common_ic_mat.': common_ic_material_ontology,
                                'selected_prod._ic_mat': selected_product_ic_material_ontology
                            }
                        )
                except Exception as e:
                    print(str(e))
                    continue


            return product_list


if __name__ == "__main__":
    app = Semantic_simlarity_calculator()
    app.root.mainloop()
```