

Article

A Semantic Model in the Context of Maintenance: A Predictive Maintenance Case Study

Gokan May ¹, Sangje Cho ^{2,*}, AmirHossein Majidirad ¹ and Dimitris Kiritsis ³

¹ Department of Mechanical Engineering, University of North Florida, Jacksonville, FL 32224, USA; gokan.may@unf.edu (G.M.); a.majidirad@unf.edu (A.M.)

² School of Engineering Technology, Purdue University, West Lafayette, IN 47907, USA

³ Department of Mechanical Engineering, EPFL (SCI-STI-DK), 1015 Lausanne, Switzerland; dimitris.kiritsis@epfl.ch

* Correspondence: cho484@purdue.edu

Abstract: Advanced technologies in modern industry collect massive volumes of data from a plethora of sources, such as processes, machines, components, and documents. This also applies to predictive maintenance. To provide access to these data in a standard and structured way, researchers and practitioners need to design and develop a semantic model of maintenance entities to build a reference ontology for maintenance. To date, there have been numerous studies combining the domain of predictive maintenance and ontology engineering. However, such earlier works, which focused on semantic interoperability to exchange data with standardized meanings, did not fully leverage the opportunities provided by data federation to elaborate these semantic technologies further. Therefore, in this paper, we fill this research gap by addressing interoperability in smart manufacturing and the issue of federating different data formats effectively by using semantic technologies in the context of maintenance. Furthermore, we introduce a semantic model in the form of an ontology for mapping relevant data. The proposed solution is validated and verified using an industrial implementation.

Keywords: predictive maintenance; Industry 4.0; semantic technologies; semantic interoperability; data federation; ontology engineering



Citation: May, G.; Cho, S.; Majidirad, A.; Kiritsis, D. A Semantic Model in the Context of Maintenance: A Predictive Maintenance Case Study. *Appl. Sci.* **2022**, *12*, 6065. <https://doi.org/10.3390/app12126065>

Received: 1 May 2022

Accepted: 13 June 2022

Published: 15 June 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The sustainability and efficiency of manufacturing systems depend on the utilization of technologies that facilitate the smart, sustainable digital factories of the future [1]. In particular, the design and deployment of predictive, prescriptive maintenance strategies that achieve increased reliability in manufacturing systems are of significant importance in fostering competitive advantage among manufacturing companies [2,3].

Manufacturing companies now face immense pressures to continuously improve the performance and reliability of their assets; at the same time, they are required to lower costs and ensure safety [4]. Furthermore, the emergence and advancement of modern, Industry 4.0 technologies—such as artificial intelligence (AI) and the industrial internet of things (IIoT)—offer companies the possibility to accurately predict machines' statuses and thus optimize maintenance activities, achieving far more than conventional reactive maintenance [5]. The future of maintenance, operations, and asset management requires manufacturing companies to move toward more proactive and optimized maintenance [6]. Further, researchers and practitioners should increase their focus on the synergies between key business processes and maintenance activities across the lifecycle of key manufacturing assets. They also need to leverage the opportunities provided by advanced maintenance solutions to optimize the cost efficiency and performance of these valuable assets [7].

The most basic and inefficient approach to maintenance is reactive maintenance, which runs assets to failure and only reacts when the equipment breaks down. Hence, this approach is often implemented with manufacturing assets that have minimal replacement

or repair costs. Such assets are not critical and do not have any significant or immediate impact on plant availability or safety [8]. The second level of maintenance is preventive maintenance (PM), which follows a fixed maintenance schedule with predefined time intervals coupled with best practices and the recommendations of manufacturers [9]. Condition-based maintenance monitors the physical condition of manufacturing equipment and operates when key and measurable performance metrics indicate critical issues before their occurrence [10]. Finally, the predictive maintenance (PdM) strategy studied in this research applies to more critical, complex assets and provides a data-driven analysis to predict a piece of equipment's operating profile during different process conditions [11]. PdM thus uses data records and Industry 4.0 technologies and methods to determine the underlying issues that affect the reliability and performance of manufacturing assets [12]. To improve scheduled PM, Van Staden et al. designed a sequential "predict, then optimize" approach to bridge PM with a predictive approach [13]. In the same context, a novel hybrid approach is implemented to assess and test real-world and synthesized information through a combination of statistical and symbolic AI to lay the foundations for Industry 4.0 PdM [14].

The PdM functions by accessing data relevant to the status and condition of machinery and focuses on the knowledge to be gained from the data [15,16]. A PdM mechanism hence aims to improve the reliability of production assets by identifying potential problems and providing early warning of issues and critical events before the occurrence of catastrophic failures in the production run [17,18].

Advanced technologies in modern industry collect massive volumes of data from a plethora of sources, such as processes, machines, components, and documents [19]. This also applies to the case of PdM. Hence, to provide access to all these data in a standard and structured way, researchers and practitioners must design and develop a semantic model of maintenance entities to build a reference ontology for maintenance [20]. Semantic enrichment can be achieved by exploring and filtering cross-sectoral information related to maintenance [21]. Semantic interoperability is the capability of information systems to unambiguously exchange data with shared meaning [22].

Scope of the Paper

In response to these challenges, this paper addresses the issue of federating different data formats effectively in the maintenance context using semantic technologies. Accordingly, we propose a method and validate our approach by designing and implementing a maintenance ontology for a case study: a European manufacturing company.

2. Motivation for Implementing Semantic Technologies toward Industry 4.0

During the last two decades, manufacturing problems have become more diverse and complicated [23]. Where the contexts of data in obsolete systems are ambiguous or lost, data-management challenges are as follows [24]:

- The documents and guidelines depicting the use of datasets may be corrupted or lost.
- Often, a data description disaccords with its field name.
- It is hard to manage tabular datasets or update their data schemas.

Even if a new database or table is created to overcome these challenges, there remains a problem with data mapping. When field names are updated, it is difficult to retrieve data in the previous database or table. Moreover, data instances can have different names because technical terminology depends on the services and functions provided. For instance, global and multicultural companies may face a multilingual information problem, depending on the locations and languages.

To solve the data-mapping problem automatically through communication between the old and new datasets, a company should make extra efforts to understand the structure of the data schema and the meaning of the data. Sometimes, it is impossible to understand these aspects with the provided information. Furthermore, depending on the data source, datasets are stored in different formats that need to be reflected in the automatic method.

Therefore, creating an automatic process may become a highly demanding task in terms of both time and resources. In this regard, Mami et al. (2020) proposed a graphic-based user interface to help non-experts efficiently interact with data [25]. They employed a SPARQL protocol and resource description framework query language (SPARQL) query to extract surface mount technology (SMT) data using triples.

Today, the emergence and advancements of Industry 4.0 technologies are leading to a paradigm shift in manufacturing. Manufacturing applications have been widened, and manufacturing companies are trying to seamlessly connect their manufacturing assets [26]. In addition, their assets are creating big data that captures their behaviors [27]. However, the challenges mentioned above complicate data management since manufacturing companies first need to spend time and resources to perform unwieldy research to comprehend the data schema and formats.

Semantic technologies provide certain capabilities to machines and systems. Using these technologies, machines can do more beneficial work, and systems facilitate reliable communication between machines. Semantic technology facilitates the capture of data's meaning. To cope with the challenges mentioned above, semantic technology provides clear and explicit meanings for data instances, an ontology as a semantic model, standard metadata, and reference of semantics. It enables machines to share a common understanding with humans. The ontology provides certain rules to describe data instances and their labels; therefore, following the rules of the ontology, all data instances can have meaning and context. In other words, semantic technology facilitates context-awareness by linking a data instance to a predefined class in an ontology or another instance, or both. The context of data can be clearly and explicitly stated because an ontology provides strong terminology based on natural language. For this reason, it enables convenient data storage and retrieval.

It is of paramount importance that an ontology provides standard meanings and rules for data. Particularly in an Industry 4.0 environment, it is necessary for multiple systems to communicate with each other. Generally, each system has its own format to describe data; hence, interoperability is essential and critical to overcoming issues caused by the interaction of a variety of formats. An ontology provides appropriate locations to store data, as it is a graphical representation of domain knowledge, including explicit definitions of terms. In addition, it is easy to retrieve data through an ontology, even for non-experts, since it describes domain knowledge in natural language.

3. State of the Art on Ontologies

3.1. *Ontology-Based Knowledge Representation and Relevant Efforts for Maintenance*

Ontology engineering presents methods and principles that are used when building ontologies [28]. This is an important field of study, as ontologies have a crucial role in knowledge management: they enable knowledge sharing and knowledge processing between domain experts and users [29]. Hence, ontologies enable a common understanding of any domain that is communicated across application systems and among people [30]. There have been many endeavors to address the issue of knowledge management and representation in the maintenance domain. In this regard, James et al. (2017) proposed a knowledge-management mechanism for system failures in automobiles in the form of an ontological model to improve maintenance processes in the automotive industry [31]. Wan et al. (2017) mapped decision-making and knowledge-management processes through an ontology and highlighted the collaboration of various factors in managing the lifecycle of a machine tool [32]. Liang (2020) proposed and implemented an ontology-based approach for achieving semantic interoperability across several systems in a collaborative manufacturing environment [33]. This system permits knowledge management and effective decision-making in automotive troubleshooting. The most recent work, by Polenghi et al. (2021), focused on prognostics and health management on the shop-floor level, utilizing ontologies to effectively integrate product-related and process-related data [34]. This facilitates a joint decision-making mechanism for maintenance and production.

3.2. Semantic Interoperability in Smart Manufacturing

Semantic interoperability is required to integrate information from heterogeneous sources of data and to interpret this information in a standardized way [35]. Several scholars have focused on achieving semantic interoperability in Industry 4.0. For example, the work of Adamczyk et al. (2020) contributes to the field by developing a semantically interoperable manufacturing system [36]. It uses ontologies to overcome semantic barriers in the representation, sharing, and reuse of knowledge and information across different phases of manufacturing processes. Weichhart et al. (2021) highlighted semantic interoperability as the key factor for supporting system-of-systems implementations [37]. They illustrate the need for interoperability in system-of-systems, as opposed to integration in a single system, focusing specifically on cyber-physical manufacturing systems. A recent study by Margaria and Steffen (2021) overcame the challenge of semantic interoperability by establishing a model-driven, digital thread platform that proposes reusable and generalizable techniques and methods to cover the integration space (domain by domain) in cyber-physical production systems [38].

3.3. Semantic Technologies and Ontologies for Predictive Maintenance

A PdM platform should be based on the continuous monitoring of the behavior of manufacturing assets and the utilization of several data sources across different levels of a manufacturing system (e.g., component, machinery, production system, product). As state-of-the-art ontologies model these problems independently, Canito et al. (2021) investigated existing ontologies in the domain of PdM and proposed extensions to bridge the gaps with machine learning [39]. Cho et al. (2019) addressed the issue of data management and integration in PdM and proposed a new method to effectively integrate data for Industry 4.0 applications via semantic technologies [19]. Cao et al. (2019) provided an approach based on ontologies to support and foster PdM applications in the manufacturing industry using a combination of semantic technologies and fuzzy clustering [40]. In a later work, Cao et al. (2020) introduced another method: a combination of semantics and data mining to predict failures in industrial machines [41]. They developed a PdM ontology for manufacturing, including a rule-based extension to create a formal representation of the predictive results. One of the latest research studies by Polenghi et al. (2021) highlighted the importance of cross-functional knowledge management [34]. They also stressed the need for and significance of both process- and product-related information for joint decision-making in maintenance and production and demonstrate the role and importance of ontologies for this purpose in smart factories.

3.4. Research Gap

As highlighted here and in the pertinent literature, to date, there have been numerous studies combining the domains of PdM and ontology engineering. However, earlier works that focused on semantic interoperability to exchange data with standardized meanings did not fully leverage the potential of data federation to further elaborate these semantic technologies. Therefore, the gap to be filled by this paper is the issue of federating various data formats across different manufacturing phases by using semantic technologies in the PdM context. In addition, we implemented a PdM ontology in a real-life manufacturing scenario.

4. Two-Cycle Methodology to Build a Semantic Model

To build an ontology to use as a semantic model, it is important to apply a proper methodology. The explicitness of the context is a primary feature of an ontology: it should have a very clear scope and specific boundaries. For this purpose, we propose a two-cycle methodology that includes a vocabulary cycle and a context cycle (Figure 1). Figure 1 shows how the two cycles interact. Each cycle includes five manual processes. This methodology, therefore, creates a solid, explicit, clear foundation for an ontology.

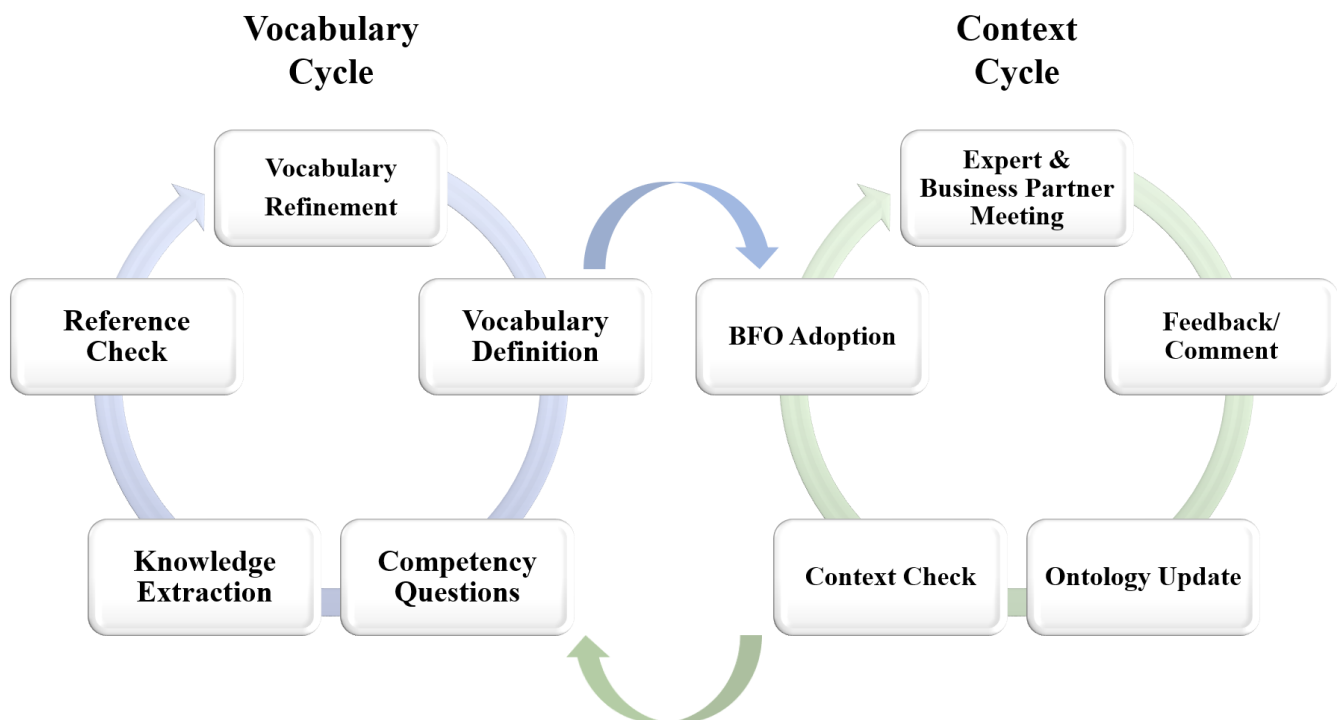


Figure 1. Two-cycle methodology.

4.1. The Vocabulary Cycle

To achieve explicitness for a semantic model in the context of maintenance, Grüninger and Fox (1995) introduced a methodology to create an ontology as an enterprise model [42]. They listed a set of carefully characterized problems, named competency questions, and used them as a foundation to build an ontology. They intended this ontology to solve the competency questions. Today, it is one of the most widely used methods to build an ontology. Competency questions are a way to determine the scope of an ontology and to sketch a list of questions to be answered [42]. These are forms of the questions in the domain of interest used to easily define the scope and boundaries of an ontology.

In the Z-BRE4K project, competency questions were created after repeated discussions with three business owners. The businesses involved a lighthouse manufacturing process producing automotive chassis, cold forming tooling, and molding machines. The list of competency questions for this study is as follows:

- In each use case, which machines are considered?
- For each use case, what stakeholders and/or actors are considered?
- For each machine, what is the bill of materials (represented as a hierarchy diagram)?
- In a specific use case, which component is critical for the management of assets?
- For each critical component, what are the failure modes?
- For each failure mode, what are the effects?
- For each failure mode, what is the criticality?
- Before the failure, what actions are needed?
- After the failure, what actions are needed?
- Which sensors are required for each critical component?
- Which critical components are associated with a sensor?
- For any sensor, what signals are collected?

These questions can be used to check the following questions: Does the ontology include sufficient information in response to these questions? Is a specific level of detail or representation of a particular area required to answer the questions? The list of competency questions does not need to be exhaustive and can be considered a sketch [43].

From these competency questions, entities are easily captured. Since they will be the basis of the taxonomy of the ontology, competency questions are useful to recognize the relationships between entities. The easiest way to capture entities is to find nouns in the competency questions. For instance, we can easily find the nouns *failure_modes* and *critical_components* in the question, “For each critical component, what are the failure modes?” To answer this question, the relationship between *failure_modes* and *critical_components* is needed. Based on this competency question, we designed the logical diagram in Figure 2 with the following contexts:

- Each *System* as a maintainable item plays its own *Role*, and the *Role of the System* inheres in *Maintenance_Process_Definition*
- Each *System* consists of the *System_part* and the *Embedded Data-Source_Component*
- The *Embedded Data-Source_Component* generates the *Source_Data*
- Each *System_part* has a *Quality*, a *Failure_Mode*, and an *Indicator* showing performance
- Each *Failure_Mode* has a *Failure_Symptom* and an *Indicator* presenting its risk, probability, and so on
- Depending on the condition, the *System* requires *Maintenance*
- The *Product_Provider* provides the *System*
- The *Maintenance_Service_Provider* provides the *Maintenance_Process_Definition*
- The *Business_Customer* owns the *System*

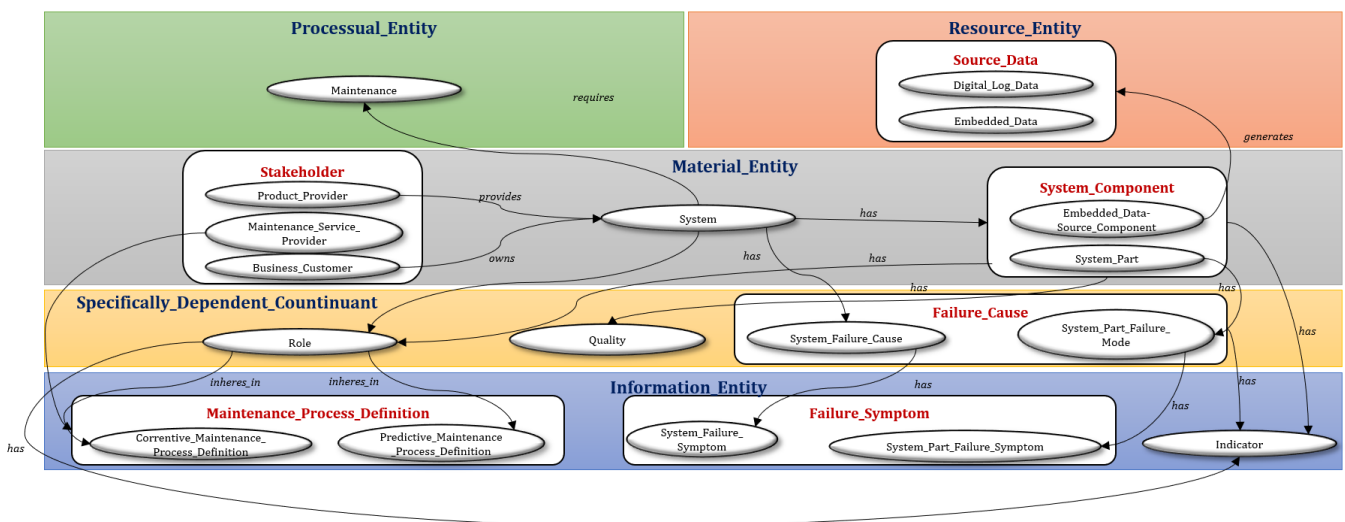


Figure 2. Logical diagram of the maintenance semantic model.

After identifying the entities and their relationships from the knowledge extraction process, they are compared to the reference ontologies. The identified entities and their relationships are elaborated to reflect the same philosophy and to enable communication with the references. Thus, to suit the requirements of end-users, the ontology in this paper can be merged with one or more reference ontologies. In this phase, the similar contexts in the references are carefully and manually checked. The reference ontologies in this task are the basic formal ontology (BFO) [44], the product-service system (PSS) ontology [45], and the product lifecycle (PLC) ontology. These were selected because they all have the same philosophy as the BFO, and the PLC and PSS ontologies were created in reference to each other.

For an ontology to be a generic and applicable semantic model, the names of entities should use the standard vocabulary in the scope. All of the vocabulary should be explicitly defined and refined (this occurs during vocabulary definition and refinement), and definitions should be described in the ontology. After vocabulary refinement, all of the selected terms are given definitions. In this study, the definitions were made based on the

definitions in the reference ontologies. If a term was missing in the reference ontologies, we have elaborated it, so its definition matches existing contexts in the references.

4.2. The Context Cycle

BFO is a formal ontology framework developed by Smith and his associates [44]. BFO comprises two disjoint ontologies: continuant and occurrent ontology. The continuant ontology includes continuant entities, such as three-dimensional enduring objects; occurrent ontology covers entities with four dimensions that unfold during a time interval (e.g., life, smiling, sending an email). Based on the BFO, we established ontology design principles as follows:

- Single nouns should be used (except “data”).
- Acronyms should be avoided.
- Univocity of terms and relational expression should be ensured.
- General should be distinguished from particular.
- All non-root terms should be provided with their definitions.
- Key features should be used to define terms and avoid circularity.
- The most generic terms of the domain should be defined first.
- Simpler terms should be used in comparison to the term being defined (to ensure intelligibility).
- Creating terms for universals through logical combinations should be avoided.
- The ontology should be structured around an “is_a” hierarchy, and completeness of “is_a” must be ensured.
- Single inheritance should be ensured.

The authors held biweekly meetings to address feedback and comments from domain experts and business partners. In these meetings, all participants intensely discussed terminology in the ontology and its definitions. Then, the relationships were checked to ensure alignment with the reference ontologies.

Not only the context cycle but also vocabulary is iterative. Consequently, competency questions, the entities, their relationships, and definitions were updated by iterations of the two-cycle methodology. Utilizing the stated methodology, a maintenance ontology was constructed through iterative cycles. This was applied to a specific project, providing a platform for PdM that led to improvement in the useful remaining life of the production systems.

5. Maintenance Ontology and Implementation

5.1. Entities of the Maintenance Ontology

To ontologically represent the structure of a target system and its failure mechanism in a standard way, the maintenance ontology refers to the Failure Mode, Effects and Criticality Analysis (FMECA) methodology, a widely recognized tool to study and analyze the reliability of a process [46]. The International Electrotechnical Commission published IEC 60812 to explain how the FMECA variant is planned, performed, documented, and maintained for general use [47]. In IEC 60812, the system structure is represented as follows:

- System breakdown to major subsystems that include functional relationships
- Properly tagged inputs and outputs and identification numbers to consistently reference each subsystem
- Redundancies, alternative signal paths, and key engineering features that facilitate protection against system failures

Referring to IEC 60812, a logical diagram (Figure 3) was created as part of the implementation of our Z-BRE4K project [48]. This logical diagram includes failure mechanism entities, physical entities, and information entities. Failure mechanism entities describe how a component’s failure affects system failure and related failure symptoms. Physical entities depict the physical structure of maintainable items and embedded data-source components. Maintainable items are the physical entities that form a part or an assembly of parts—such as a part, module, subsystem, or system—that are at the lowest level in

the equipment hierarchy during maintenance. ISO 14224:2006 [49] Appendix A provides examples of maintainable items for a variety of equipment. Information entities represent information and data generated from advanced enabling technologies that can be used in management.

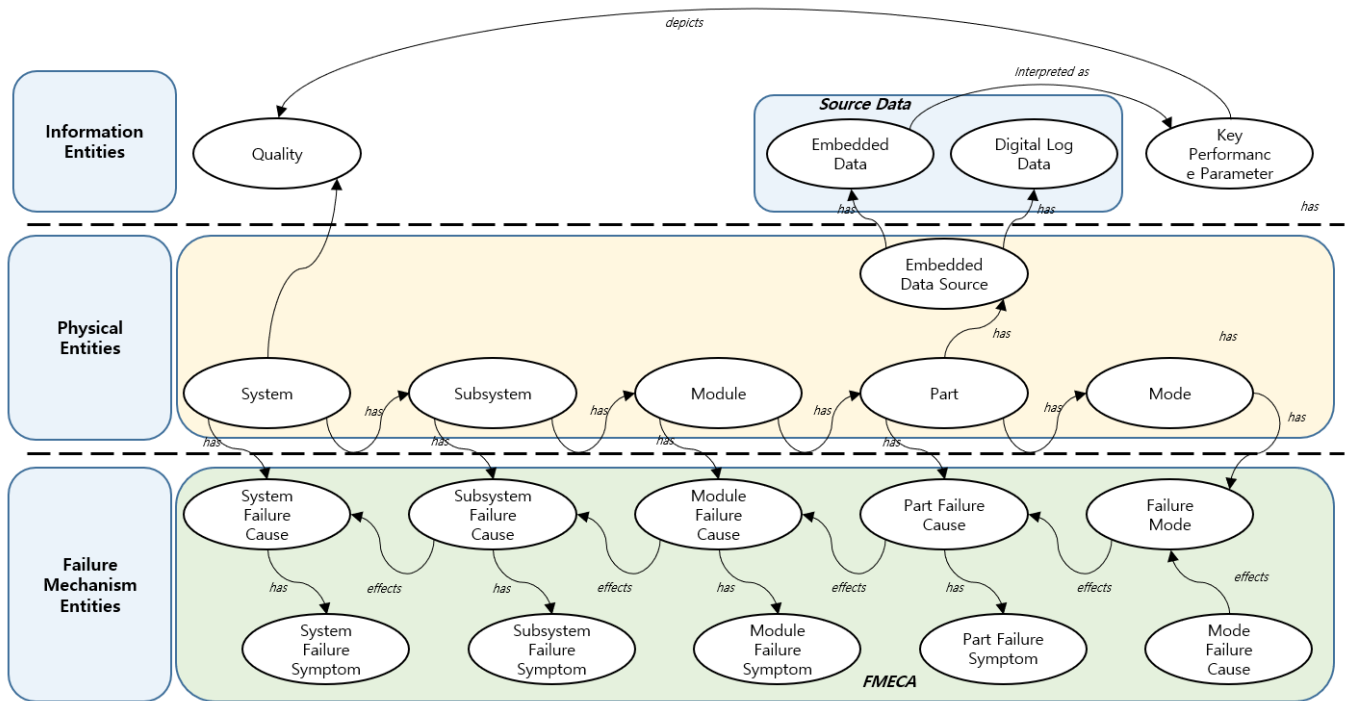


Figure 3. Logical diagram of maintainable items and failure mechanisms.

The maintenance ontology was created through iterative cycles of the methodology we proposed. This task was accomplished as part of the H2020 Z-BRE4K project. The Z-BRE4K project [48] provides a platform for PdM to eliminate unplanned breakdowns and hence extend the useful life of production systems.

The Z-Bre4k solution is comprised of the introduction of eight scalable strategies at component, machine, and system levels: (i) the prediction of failure based on evidence (Z-PREDICT), (ii) the early detection of current or emerging failure (Z-DIAGNOSE), (iii) the prevention of failure occurrence, accumulation, or propagation within the production system (Z-PREVENT), (iv) the estimation of the remaining useful life (RUL) of assets (Z-ESTIMATE), (v) the management of these strategies through event modeling, key performance indicator (KPI) monitoring, and real-time decision support (Z-MANAGE), (vi) the replacement, reconfiguration, re-use, retirement, and recycling of components or assets (Z-REMEDiate), (vii) the synchronization of remedial actions, production planning, and logistics (Z-SYNCHRONISE), (viii) the ongoing preservation of the safety, health, and comfort of human workers (Z-SAFETY) [12]. The Z-BRE4K ontology serves as a backbone of the Z-BRE4K platform since it enables semantic interoperability between system components.

The standardization in Z-BRE4K is based on the design of the semantic model of the maintenance entities. It is also based on the reference ontology, which provides standards-based access to all the data in the domain of interest. The Z-BRE4K ontology manages the design and implementation of the semantic model and the mechanisms for intelligent filtering. It provides a unified and all-spanning semantic model covering the multi-domain knowledge of all Z-BRE4K use cases. Therefore, Z-BRE4K achieves semantic enrichment by receiving maintenance knowledge that results from cross-sectoral knowledge exploration and filtering. Thus, it requires semantic interoperability: the ability of information systems to exchange data unambiguously with shared meanings as a standard. This section depicts the form of the Z-BRE4k ontology and its implementation within the PdM context.

5.2. Implementation of the Maintenance Ontology

The maintenance ontology describes the context based on the competency questions described in Section 4.1. It comprises physical entities, realizable entities, and information entities. Physical entities are existing three-dimensional objects, realizable entities are objects or events that can be achieved or made to happen, and information entities are knowledge collected through measurement records.

The maintenance ontology has *System*, *System_Component*, and *Embedded_Data_Source* as physical entities to describe the structure of a maintainable item. The *System* is defined as a *Material_Entity*: a universal set of a target system that can be a single or multiple machine or product. *System_Component* is a material part of the *System*. The subcomponent also can be a *System_Component*. The *Embedded_Data_Source* represents a *System_Component* that is intended to generate data: it is a type of sensor or digital device. The *Stakeholder* is also a physical entity: the human (or being) that manages a maintenance scenario.

As realizable entities, there are *Failure_Cause* and *Quality* entities. *Failure_Cause* is a defect in a process that is the underlying cause of a failure or that initiates a process leading to failure. *Quality* is a characteristic of physical entities. Failure can be a sort of *Quality*.

The information entities are *Indicator*, *Process_definition*, and *Failure_Symptom*. The *Indicator* is an observed value that indicates the level or state of an entity. *Process_definition* is the structured set of activities that are designed to accomplish a particular objective.

These three types of entities, as a foundation of classes for a maintenance ontology, were extended in the maintenance ontology (Figure 4). We present the list of classes together with their definitions in Table A1 in Appendix A.

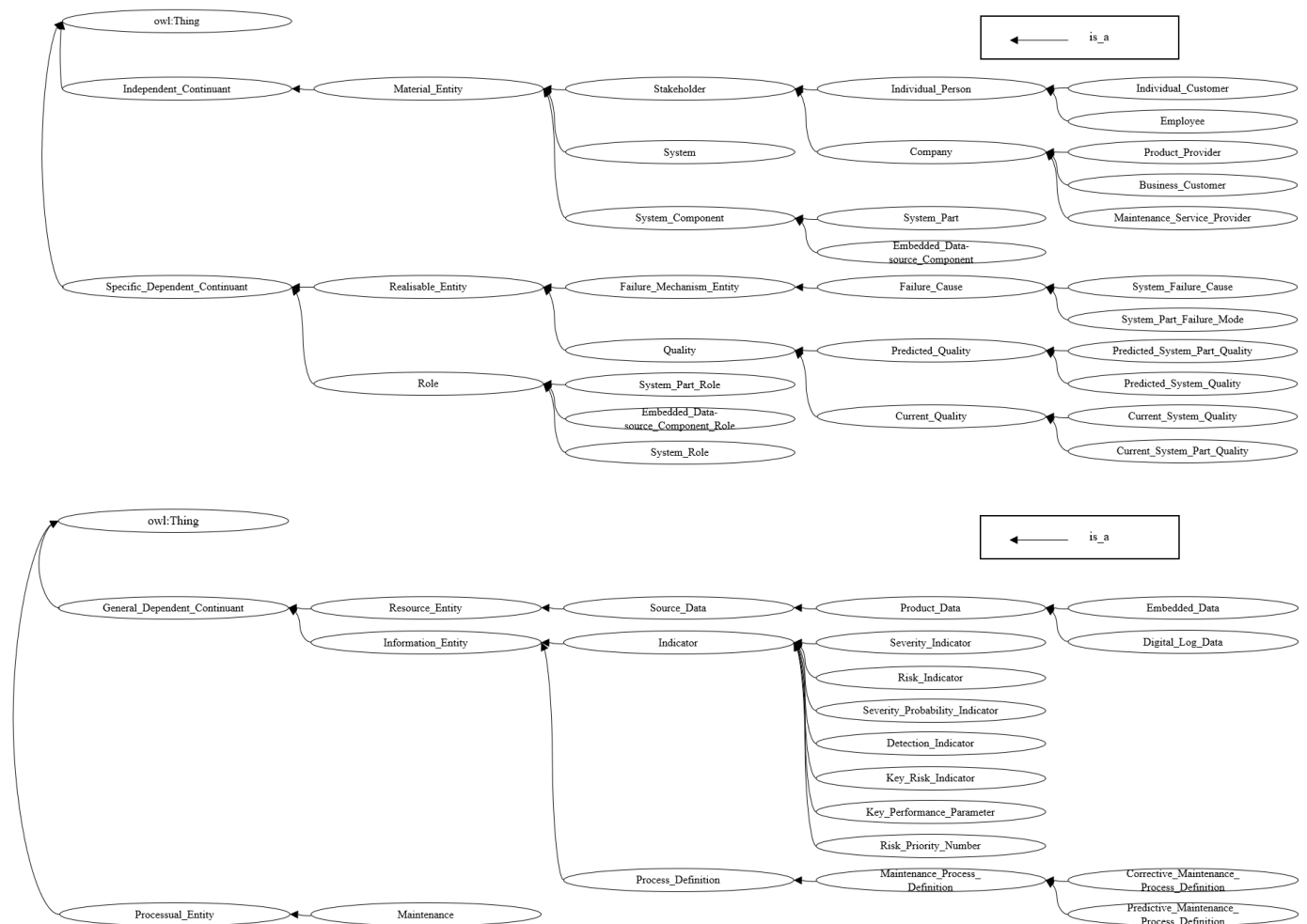


Figure 4. Z-BRE4K semantic model hierarchy.

To enhance the ontology of maintenance, this study deals with information and data from the packaging industry. The packaging company provides service to its end users for compression molding machines concerning PdM. The ontology was specialized for this industry after it was populated since this ontology is a generalization of the business scenarios, a reference semantic model. The populated ontology codifies and encapsulates explicit knowledge of the machine operating in the packaging company. In this study, we present the case of the compression molding equipment as a target system. It consists of pumps, screws, bypass valves, cooling fans, mixers, heaters, cylinders, and polymer as system components (Figure 5). Their failure modes and symptoms are described in Table A2 in Appendix A.



Compression Molding Machine

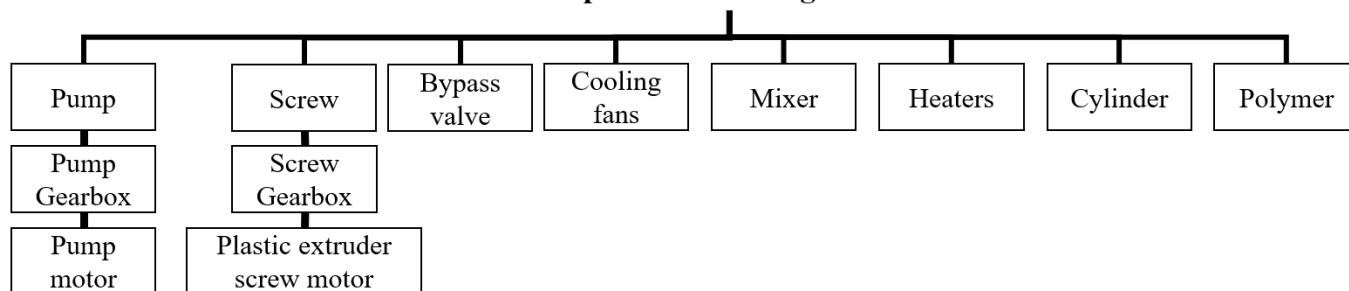


Figure 5. Structure of the compression molding machine.

According to the Z-BRE4K ontology, a structure of a target system can be presented within the class system and *system_parts*. The physical objects of a system and its parts should be instances under these classes. The notions of physical objects are described as roles, failure modes, and symptoms associated with these notions. A physical object can have a sensor as an *embedded_data_source* that generates the data properties *embedded_data*, which has a measured value, and date and time. *Embedded_data* is interpreted as *Performance_Parameters*, which are indications of quality measurement. If a PdM service detects a potential failure, it is described as a *predicted_quality* with an expected date and time. Figure 6 depicts how the Z-BRE4K ontology describes an example scenario.

This example from the packaging industry serves as a guide for users of this ontology. It provides a basic idea of how we designed the maintenance ontology so that users can understand the correct position of data in the ontology.

Since this ontology was created as part of the H2020 Z-BRE4K project, the maintenance ontology for the packaging industry was validated and tested with a semantic component on the Z-BRE4K platform named the knowledge base system (KBS). The KBS makes it possible for all data on the platform to have semantics following the rules provided by the maintenance ontology. It stores data in the form of graphs, making links between the data and providing context to the maintenance ontology. The KBS also facilitates describing the data in a triple format (Subject, Predicate, Object) by using a resource description framework (RDF). In addition, it provides semantic services, such as search, authentication, dataset, revision, and SPARQL (Figure 7). Thus, it facilitates governance of the maintenance ontology, controlling all the data on the platform and the semantic enrichment of the Z-BRE4K platform.

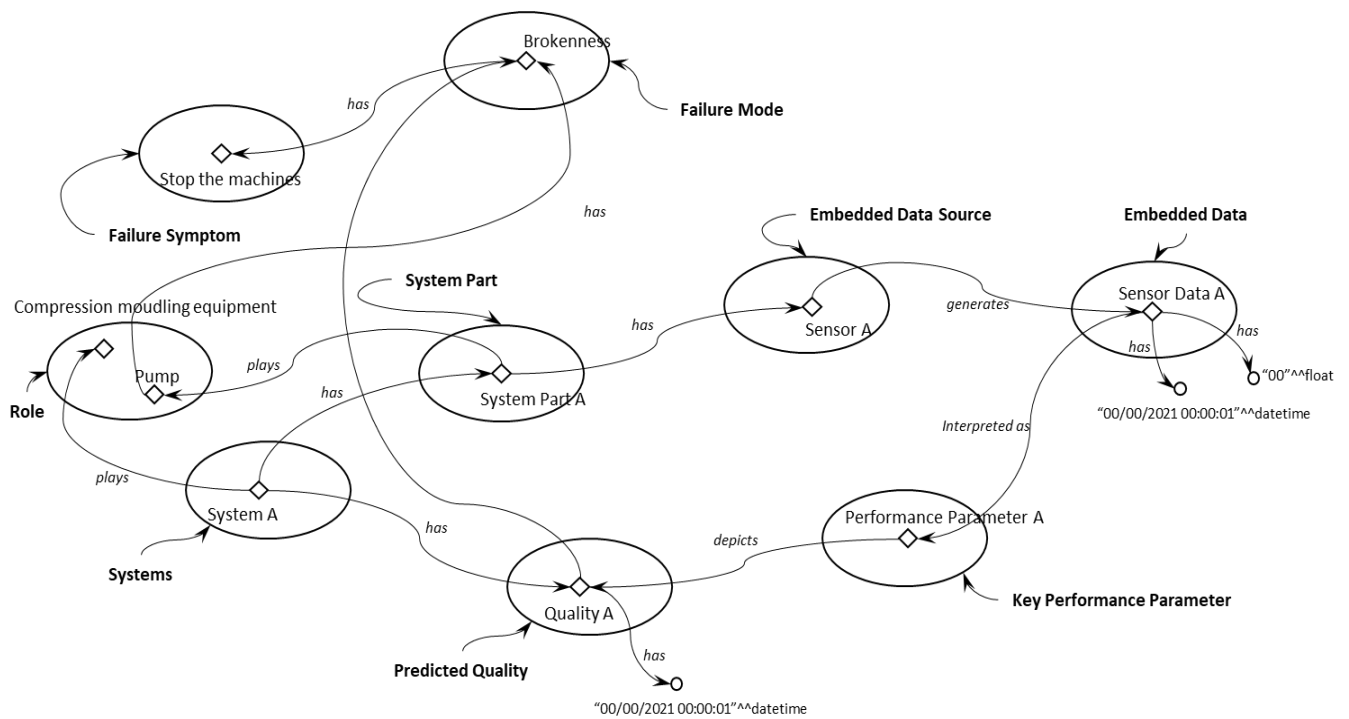


Figure 6. Semantic description of an example scenario.

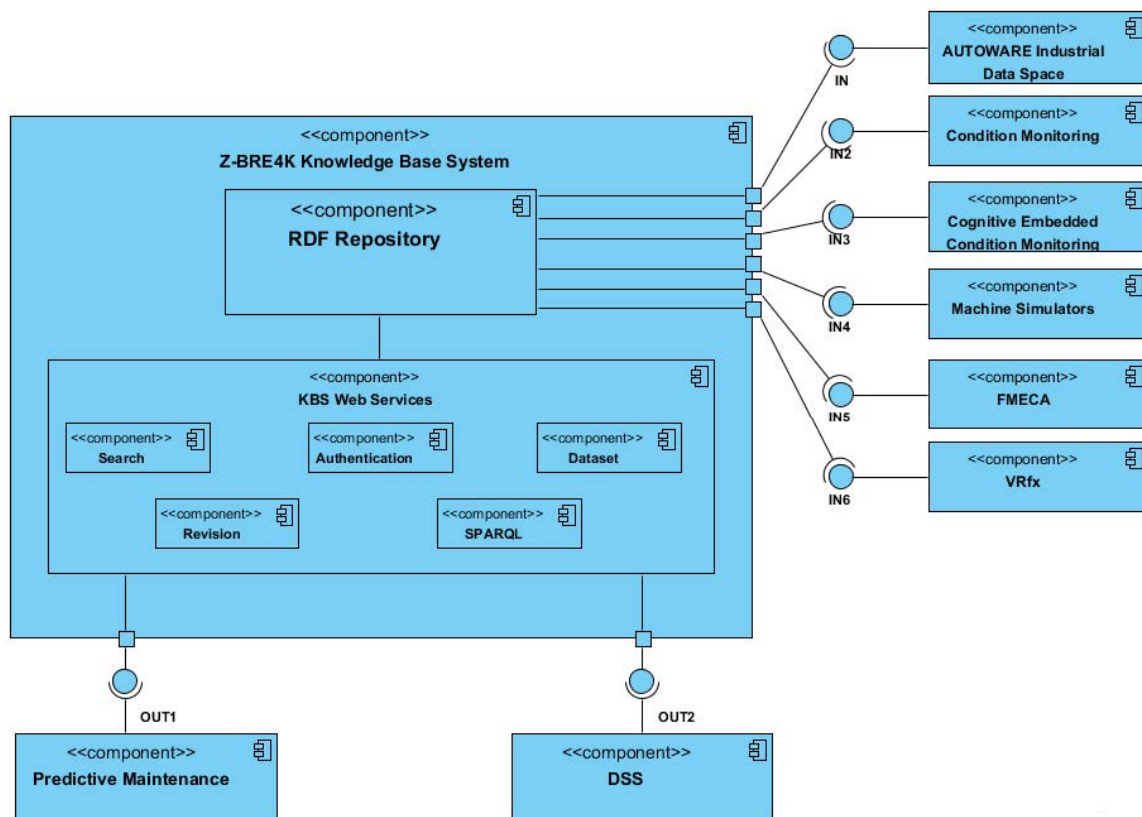


Figure 7. The knowledge base system's component diagram.

The implementation details of the Z-BRE4K semantic model as an ontology provide some insights based on the practical implementation issues that developers have encountered:

- The competency question methodology proved to be a useful tool for meeting the requirements of stakeholders in the domain field. Future efforts to develop ontologies for smart manufacturing applications should consider this method to standardize the ontology design and development process.
- The BFO framework was successfully implemented to create ontologies in the maintenance engineering domain, and other domains should follow (e.g., zero-defect manufacturing, product-service systems, energy-efficient manufacturing).
- Ontology engineering requires iterative processes. Hence, updates to the classes and their relationships from iterative feedback from end-users and technical providers in different use cases may help developers consolidate the ontology for application to further scenarios.

A preliminary assessment of the evidence-based effectiveness of combining the data and this management approach is confirmed by the successful implementation of the Z-BRE4K platform, as presented in detail by Rousopoulou et al. (2022) [50]. The Z-BRE4K system, which was developed following the principles of the semantic model provided, has contributed to industrial digitization and the creation of smart factories. It provides a generic platform as a complete solution, supporting standards-based factory connectivity, data management, various AI models' training and comparisons, live predictions, and real-time visualizations.

Finally, Table A3 (in Appendix A) presents the Z-BRE4K ontology's object properties.

6. Discussion and Concluding Remarks

A review of both research literature and industrial practices concerning ontologies shows that there are no quantitative measures or KPIs to evaluate and compare the results and effectiveness of industrial ontologies. This is one of the major limitations of ontology development studies and a potential subject for further studies. For this reason, in this section, we compare our study with the most pertinent literature. We consider various key features of design and development, such as the main principles adopted for ontology design, whether the ontology was tested in an industrial use case or in the development of a software platform, the focus of the ontology, and whether any method was followed when designing the ontology.

In this paper, we followed the design principles of the BFO and used a two-cycle methodology for semantic model design, as explained in detail in Section 4. The focus of our ontology is maintenance in the manufacturing context, and an industrial test was performed to validate the ontology that also aided the successful implementation of the Z-BRE4K platform [50].

The most pertinent research to compare to our study in terms of achievements are by Cao et al. (2022) [51], Montero-Jimenez et al. (2021) [52], Polenghi et al. (2021) [34], Emmanouilidis et al. (2020) [53], Karray et al. (2019) [54], and Cao et al. (2019) [40]. The method we followed for the development of the semantic model—the use of a two-cycle methodology, competency questions, and BFO compliance—is the strength of our model. Table 1 highlights the achievements of our research in comparison with other relevant literature.

Table 1. Comparison of our research with relevant literature.

Publication	Focus on Maintenance	Is Any Formal Method Used during the Ontology Development Process?	Is the Ontology Tested in an Industrial Case or Used for the Development of a Platform?	Is Any Specific Method Developed by the Authors for the Semantic Model Design?
This manuscript (May et al., 2022)	+	Basic Formal Ontology	+	+ Two-cycle methodology to build semantic model
Cao et al. (2022) [51]	+	Reused previously developed ontologies	+	+ Core Reference Ontology (CROS)
Montero-Jimenez et al. (2021) [52]	+	Noy and McGuinness (2001)	-	-
Polenghi et al. (2021) [34]	+	Reused previously developed ontologies	+	-
Emmanouilidis et al. (2020) [53]	+	-	+	-
Karray et al. (2019) [54]	+	Basic Formal Ontology	+	-
Cao et al. (2019) [40]	+	Noy and McGuinness (2001)	+	-

This paper reviewed the data-federation and data-management challenges of obsolete systems and proposed the use of semantic technology for Industry 4.0. Inspired by the competency questions, the two-cycle methodology was introduced to create the domain ontology. It comprises a vocabulary cycle and a context cycle to strengthen semantics through interactive updates.

The maintenance ontology was created through the two-cycle methodology. This ontology was designed considering the use together with other domain ontologies, following BFO principles. This ontology was tested and verified on the Z-BRE4K platform with KBS. As semantic services are provided by the KBS, the maintenance ontology was used for data governance as part of the platform. This article discusses interoperability at semantic and technical levels. Other key contributions were removing ambiguous terms and redundant contextual terminologies, focusing on established vocabularies, and emphasizing a structure to compartmentalize essential vocabulary. This approach could be suitable for broader adoption; however, this falls beyond the scope of this study.

Leveraging the maintenance ontology and the encapsulation of maintenance domain knowledge helps to achieve semantic enrichment for the manufacturing industry. Future work could utilize the proposed method in other relevant manufacturing practices, such as agile manufacturing, additive manufacturing, information integration in cloud manufacturing, and digital manufacturing.

Semantic interoperability in industry remains a challenge. To successfully adopt Industrial Ontologies Foundry (IOF) ontologies and adapt them to different industries, a real-world application and associated validation are crucial. The proposed ontology was verified on the stated platforms; however, the validation of this methodology on large datasets remains a topic for future research. To overcome this limiting factor in future studies and align with the vision of the current work, a large-scale testing environment could promote learning for researchers who develop and propose ontologies and for customers who use ontologies in their industries. On a more complex level, research could be conducted in industries with different geographical locations, diverse cultures, and a variety of organizational issues. This could help highlight the importance of data integration and harmonization. An interoperable ontology that considers information gathered from several stakeholders leads to operational improvements in manufacturing. As a continuation of this work, progressing beyond the needs of local industries, recruiting data engineers, and

comprehensively studying resources could further promote the development of a robust ontology scheme.

Author Contributions: Conceptualization, G.M. and S.C.; Formal analysis, S.C.; Funding acquisition, D.K.; Methodology, G.M., S.C., A.M. and D.K.; Supervision, G.M.; Validation, G.M. and S.C.; Writing—original draft, G.M., S.C. and A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the European Commission as a part of the Z-BRE4K project (Grant Agreement Number: 768869). The APC was funded by the University of North Florida.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

We present the list of classes of the ontology together with their definitions in Table A1.

Table A1. Z-BRE4K ontology classes and definitions.

Class Name	Definition
Independent_Continuant	Independent_Continuant (see BFO)
Material_Entity	Material_Entity (see BFO)
System	The System is a Material_Entity which is a universal set of the target system
System_Component	Product Component is a Material Entity which is a part of the System.
System_Part	The System_Part is a System_Component which are a modularity subset of the system
Embedded_Data-Source_Component	Embedded Data-Source Component is a Product Component that refers to product-embedded information devices/parts (e.g., sensors), which allows for (additional) services to be offered for the product.
Stakeholder	Stakeholder/ Actor is an entity that describes the humans (or beings) involved in a business scenario (development, deployment, use etc.).
Individual_Person	Individual Person is a Stakeholder indicating a separate person who is not a legal entity (i.e., not legally considered a corporate group, as for example a company).
Individual_Customer	Individual Customer is an Individual Person that buys or consumes (consumer) any product/maintenance service.
Employee	An employee is an Individual Person that works in a Company in the domain.
Company	Company is a Stakeholder that can be one or more individuals forming a legal entity (corporate body).
Business_Customer	Business Customer is a Company that buys and uses the products, and maintenance services. It is the (group/company) customer of the system, using the system for their own business.
Product_Provider	Product Provider is a Supplier/provider/vendor that both produces and sells a product or only sells it.
Maintenance_Service_Provider	Service_Provider is a Supplier/provider/vendor that offers a maintenance service.
Generally_Dependent_Continuant	Generally Dependent Continuant (see BFO)
Resource_Entity	Resource_Entity is a mean (source) that can be used for the design, development, offer and delivery of a PSS, from which benefit can be produced.
Source_Data	Source-Data is a Resource_Entity that refers to the digitalised information coming as input from the product or the service, specifically for the scope of Project.
Product_Data	Product Data are Source_Data used in the scope of the product and refer specifically to the system part.
Embedded_Data	Embedded Data is Product Data generated from embedded data-source components.
Digital_Log_Data	Digital Log Data is Product Data generated from a software element, part of the product.
Information_Entity	Information_Entity is a Resource, immaterial, describing aspects affecting Project, and referring to all the (processed) data that one can acquire knowledge from, or further process.
Indicator	The Indicator is an Information_Entity which is an observed value indicating the state or level of something
Key_Performance_Parameter	The Key_Performance_Parameter is an indicator to measure the performance of each mode at time t

Table A1. Cont.

Class Name	Definition
Key_Risk_Indicator	The Key_Risk_Indicator is an indicator of risk exposures to address potential warning events
Risk_Probability_Indicator	The Risk Probability Indicator is an indicator which is a frequency of occurrence of Failure_Effect
Severity_Indicator	The Severity Indicator is an indicator of how serious and destructive a failure mode is
Detection_Indicator	The Detection_Indicator is a numerical subjective estimation of the effectiveness of the controls to prevent or detect the cause or failure mode before the failure (Threshold?)
Risk_Indicator	The Risk is an indicator multiplying Probability Indicator by Severity Indicator of the failure
Risk_Priority_Number	The Risk Priority Number is an indicator multiplying Risk Indicator by Detection Indicator of the failure
Failure_Symptom	The Failure_Symptom is a Failure_Mechanism_Entity which is a physical feature that is regarded as indicating a condition of the failure
System_Failure_Symptom	The System_Failure_Symptom is a Failure_Symptom described in the level of System
System_Part_Failure_Symptom	The Subsystem_Failure_Symptom is a Failure_Symptom described in the level of System_Part
Process_Definition	Process Definition is a structured set of activities designed to accomplish a specific objective. For example, the Process Definition can be the manufacturing process where the equipment is manufactured, or the process where the equipment is used, or the development process where products and maintenance services are being developed etc.
Maintenance_Process_Definition	Maintenance Process Definition is a Process Definition referring to a set of non-tangible entities (activities, software modules etc.), related to the maintenance service part, which aims to satisfy a need of the end-users.
Predictive_Maintenance_Process_Definition	The Predictive_Maintenance_Process_Definition is a Maintenance_Process_Definition to prevent failures by monitoring
Correntive_Maintenance_Process_Definition	The Correntive_Maintenance_Process_Definition is a Maintenance_Process_Definition to correct failures after break down or malfunction
Specifically_Dependent_Countinuant	Specifically Dependent Continuant (See BFO)
Realisable_Entity	Realisable Entity (See BFO)
Failure_Mechanism_Entity	Failure_Mechanism_Entity
Failure_Cause	The Failure_Cause is a Failure_Mechanism_Entity which is a defect in design, process, quality, or part application, which are the underlying cause of a failure or which initiate a process which leads to failure.
System_Failure_Cause	The System_Failure_Cause is a Failure_Cause described in the level of System
System_Part_Failure_Mode	The System_Part_Failure_Mode is a Failure_Cause described in the level of System_Part
Quality	Quality (See BFO)
Current_Quality	The Current_Quality is a Quality at time t. If they inhere in an entity at all, they are fully exhibited or manifested or realized in that entity.
Current_System_Quality	The Current_Quality is a Quality of a System entity at time t.
Current_System_Part_Quality	The Current_System_Part_Quality is a Quality of a System_Part entity at time t.
Predicted_Quality	The Predicted_Quality is a predicted Quality at future time t.
Predictied_System_Quality	The Predictied_System_Quality is a Quality of a System entity at future time t.
Predictied_System_Part_Quality	The Predictied_System_Part_Quality is a Quality of a System_Part entity at future time t.
Role	(See BFO)
System_Role	borne by a group of one or more System(s) by Maintenance_Process_Definition
System_Part_Role	borne by a group of one or more System_Part(s) by Maintenance_Process_Definition
Embedded_Data-Source_Component_Role	borne by a group of one or more Embedded_Data-Source_Component(s) by Maintenance_Process_Definition
Processual_Entity	See BFO
Maintenance	Maintenance is a Process made for, and driven by customers, with economic value, and refers to the maintenance Offer entity in conjunction with time t.

In this study, we present the case of the compression molding equipment as a target system. Its components, failure modes, and symptoms are described in Table A2.

Table A2. System Components, Failure Modes, Failure Symptoms.

System Components	Description of Failure Mode	Failure Symptom
Plastic extruder screw motor	motor is worn	stop the machines (no polymer provided in machine)
	motor bearing is damaged	local abnormal heating local abnormal noise/vibrations
Pump motor	motor is worn	stop the machines (no polymer provided in machine)
	motor bearing is damaged	local abnormal heating local abnormal noise/vibrations
Screw gearbox	Wear	local abnormal noise/vibration
	broken	stop the machines (no polymer provided in machine)
Pump gearbox	Wear	local abnormal noise/vibration
	broken	stop the machines (no polymer provided in machine)
Pump	wear (teeth and bearing)	abnormal flow rate and pressure local abnormal noise/vibration
	broken	stop the machines (no polymer provided in machine)
Polymer	wrong composition	abnormal process conditions (flow rate, temperatures, pressures)
	presence of humidity within the polymer	abnormal process conditions (flow rate, temperatures, pressures)
Bypass valve	Block	stop the machines (no polymer provided in machine)
	starts sticking	increase in the switching time incomplete switching of the valve increase in the pump outbound pressure
Cooling fans	motor is worn	abnormal temperature regulations (no airflow provided)
	motor bearing is damaged	abnormal temperature regulations (different amount of airflow provided)
Mixer	broken	thermal drift of the melted polymer irregular color mixing
Heaters	burns itself	abnormal temperature regulations (different heating provided)
Screw	Wear	increase in the screw speed abnormal melt pressure fluctuation abnormal flow rate fluctuation abnormal melt temperature abnormal temperature regulations
		increase in the screw speed; abnormal melt pressure fluctuation; abnormal flow rate fluctuation; abnormal melt temperature; abnormal temperature regulations
Cylinder	wear	increase in the screw speed; abnormal melt pressure fluctuation; abnormal flow rate fluctuation; abnormal melt temperature; abnormal temperature regulations

Table A3 presents the Z-BRE4K ontology's object properties.

Table A3. Z-BRE4K Ontology Object Properties.

Object Property	Domain	Range
has_System_Part	System	System_Part
has_System_Failure_Cause	System_Role	System_Failure_Cause
has_System_Part_Failure_Mode	System_Part_Role	System_Part_Failure_Mode
effects_System_Failure_Cause	System_Part_Failure_Mode	System_Failure_Cause
has_System_Failure_Symptom	System_Failure_Cause	System_Failure_Symptom
has_System_Part_Failure_Symptom	System_Part_Failure_Mode	System_Part_Failure_Symptom
has_Embedded_Data-Source_Component_of_System_Part	System_Part	Embedded_Data-Source_Component
generates_Embedded_Data	Embedded_Data-Source_Component	Embedded_Data
inherits_in_System_of_Current_System_Quality	Current_System_Quality	System
inherits_in_System_Part_of_Current_System_Part_Quality	Current_System_Part_Quality	System_Part

Table A3. Cont.

Object Property	Domain	Range
<code>inheres_in_System_of_Predicted_System_Quality</code>	Predicted_System_Quality	System
<code>inheres_in_System_Part_of_Predicted_System_Part_Quality</code>	Predicted_System_Part_Quality	System_Part
<code>has_System_Failure_Cause_of_Current_System_Quality</code>	Current_System_Quality	System_Failure_Cause
<code>has_System_Part_Failure_Mode_of_Current_System_Part_Quality</code>	Current_System_Part_Quality	System_Part_Failure_Mode
<code>has_System_Failure_Cause_of_Predicted_System_Quality</code>	Predicted_System_Quality	System_Failure_Cause
<code>has_System_Part_Failure_Mode_of_Predicted_System_Part_Quality</code>	Predicted_System_Part_Quality	System_Part_Failure_Mode
<code>has_Detection_Indicator_of_System_Failure_Cause</code>	System_Failure_Cause	Detection_Indicator
<code>has_Detection_Indicator_of_System_Part_Failure_Mode</code>	System_Part_Failure_Mode	Detection_Indicator
<code>has_Key_Performance_Parameter_of_System</code>	System	Key_Performance_Parameter
<code>has_Key_Performance_Parameter_of_System_Part</code>	System_Part	Key_Performance_Parameter
<code>has_Key_Risk_Indicator_of_System_Failure_Cause</code>	System_Failure_Cause	Key_Risk_Indicator
<code>has_Key_Risk_Indicator_of_System_Part_Failure_Mode</code>	System_Part_Failure_Mode	Key_Risk_Indicator
<code>has_Risk_Probability_Indicator_of_System_Failure_Cause</code>	System_Failure_Cause	Risk_Probability_Indicator
<code>has_Risk_Probability_Indicator_of_System_Part_Failure_Mode</code>	System_Part_Failure_Mode	Risk_Probability_Indicator
<code>has_Severity_Indicator_of_System_Failure_Cause</code>	System_Failure_Cause	Severity_Indicator
<code>has_Severity_Indicator_of_System_Part_Failure_Mode</code>	System_Part_Failure_Mode	Severity_Indicator
<code>inheres_in_Risk_Probability_Indicator</code>	Risk_Indicator	Risk_Probability_Indicator
<code>inheres_in_Severity_Indicator</code>	Risk_Indicator	Severity_Indicator
<code>inheres_in_Risk_Indicator</code>	Risk_Priority_Number	Risk_Indicator
<code>inheres_in_Detection_Indicator</code>	Risk_Priority_Number	Detection_Indicator
<code>requires_Maintenance_of_System</code>	System	Maintenance
<code>requires_Maintenance_of_System_Part</code>	System_Part	Maintenance
<code>requires_Predictive_Maintenance_Process_Definition_of_System_Failure_Symptom</code>	System_Failure_Symptom	Predictive_Maintenance_Process_Definition
<code>requires_Predictive_Maintenance_Process_Definition_of_System_Part_Failure_Symptom</code>	System_Part_Failure_Symptom	Predictive_Maintenance_Process_Definition
<code>requires_Correntive_Maintenance_Process_Definition_of_System_Failure_Symptom</code>	System_Failure_Symptom	Correntive_Maintenance_Process_Definition
<code>requires_Correntive_Maintenance_Process_Definition_of_System_Part_Failure_Symptom</code>	System_Part_Failure_Symptom	Correntive_Maintenance_Process_Definition
<code>provides_Predictive_Maintenance_Process_Definition</code>	Maintenance_Service_Provider	Predictive_Maintenance_Process_Definition
<code>provides_Correntive_Maintenance_Process_Definition</code>	Maintenance_Service_Provider	Correntive_Maintenance_Process_Definition
<code>provides_System</code>	Product_Provider	System
<code>owns_System</code>	Business_Customer	System
<code>owns_System</code>	Individual_Customer	System
<code>works_in_Product_Provider</code>	Employee	Product_Provider
<code>works_in_Maintenance_Service_Provider</code>	Employee	Maintenance_Service_Provider
<code>has_System_Role</code>	System	System_Role
<code>has_System_Part_Role</code>	System_Part	System_Part_Role
<code>has_Embedded_Data-Source_Component_Role_of_Embedded_Data-Source_Component</code>	Embedded_Data-Source_Component	Embedded_Data-Source_Component_Role
<code>inheres_in_Predictive_Maintenance_Process_Definition_of_System_Role</code>	System_Role	Predictive_Maintenance_Process_Definition
<code>inheres_in_Predictive_Maintenance_Process_Definition_of_System_Part_Role</code>	System_Part_Role	Predictive_Maintenance_Process_Definition
<code>inheres_in_Predictive_Maintenance_Process_Definition_of_Embedded_Data-Source_Component_Role</code>	Embedded_Data-Source_Component_Role	Predictive_Maintenance_Process_Definition
<code>inheres_in_Correntive_Maintenance_Process_Definition_of_System_Role</code>	System_Role	Correntive_Maintenance_Process_Definition
<code>inheres_in_Correntive_Maintenance_Process_Definition_of_System_Part_Role</code>	System_Part_Role	Correntive_Maintenance_Process_Definition
<code>consists_of_System_Part_Role_of_System_Role</code>	System_Role	System_Part_Role
<code>consists_of_Embedded_Data-Source_Component_Role_of_System_Role</code>	System_Role	Embedded_Data-Source_Component_Role
<code>realized_in_Embedded_Data-Source_Component_Role_of_System_Part_Failure_Mode</code>	System_Part_Failure_Mode	Embedded_Data-Source_Component_Role

References

- May, G.; Stahl, B.; Taisch, M. Energy management in manufacturing: Toward eco-factories of the future—A focus group study. *Appl. Energy* **2016**, *164*, 628–638. [\[CrossRef\]](#)
- Selcuk, S. Predictive maintenance, its implementation and latest trends. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2017**, *231*, 1670–1679. [\[CrossRef\]](#)
- Mobley, R.K. *An Introduction to Predictive Maintenance*, 2nd ed.; Butterworth-Heinemann: Amsterdam, The Netherlands; New York, NY, USA, 2002.
- Zonta, T.; da Costa, C.A.; Righi, R.D.R.; de Lima, M.J.; da Trindade, E.S.; Li, G.P. Predictive maintenance in the Industry 4.0: A systematic literature review. *Comput. Ind. Eng.* **2020**, *150*, 106889. [\[CrossRef\]](#)
- Carvalho, T.P.; Soares, F.A.A.M.N.; Vita, R.; Francisco, R.D.P.; Basto, J.; Alcalá, S.G.S. A systematic literature review of machine learning methods applied to predictive maintenance. *Comput. Ind. Eng.* **2019**, *137*, 106024. [\[CrossRef\]](#)
- Ayvaz, S.; Alpay, K. Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time. *Expert Syst. Appl.* **2021**, *173*, 114598. [\[CrossRef\]](#)
- Roda, I.; Macchi, M. Maintenance concepts evolution: A comparative review towards Advanced Maintenance conceptualization. *Comput. Ind.* **2021**, *133*, 103531. [\[CrossRef\]](#)
- Cheng, J.C.; Chen, W.; Chen, K.; Wang, Q. Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms. *Autom. Constr.* **2020**, *112*, 103087. [\[CrossRef\]](#)
- Wan, J.; Tang, S.; Li, D.; Wang, S.; Liu, C.; Abbas, H.; Vasilakos, A.V. A Manufacturing Big Data Solution for Active Preventive Maintenance. *IEEE Trans. Ind. Inform.* **2017**, *13*, 2039–2047. [\[CrossRef\]](#)
- Mourtzis, D.; Vlachou, E. A cloud-based cyber-physical system for adaptive shop-floor scheduling and condition-based maintenance. *J. Manuf. Syst.* **2018**, *47*, 179–198. [\[CrossRef\]](#)
- Cho, S.; May, G.; Kiritsis, D. A Predictive Maintenance Approach Toward Industry 4.0 Machines. In *Engineering Assets and Public Infrastructures in the Age of Digitalization, Lecture Notes in Mechanical Engineering*; Liyanage, J.P., Amadi-Echendu, J., Mathew, J., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 646–652. [\[CrossRef\]](#)
- May, G.; Kyriakoulis, N.; Apostolou, K.; Cho, S.; Grevenitis, K.; Kokkorikos, S.; Milenkovic, J.; Kiritsis, D. Predictive Maintenance Platform Based on Integrated Strategies for Increased Operating Life of Factories. In *Advances in Production Management Systems. Smart Manufacturing for Industry 4.0, IFIP Advances in Information and Communication Technology*; Moon, I., Lee, G.M., Park, J., Kiritsis, D., von Cieminski, G., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 279–287. [\[CrossRef\]](#)
- Van Staden, H.E.; Deprez, L.; Boute, R.N. A dynamic “predict, then optimize” preventive maintenance approach using operational intervention data. *Eur. J. Oper. Res.* **2022**, *302*, 1079–1096. [\[CrossRef\]](#)
- Cao, Q.; Zanni-Merk, C.; Samet, A.; Reich, C.; Beuvron, F.D.B.D.; Beckmann, A.; Giannetti, C. KSPMI: A Knowledge-based System for Predictive Maintenance in Industry 4.0. *Robot. Comput. Manuf.* **2022**, *74*, 102281. [\[CrossRef\]](#)
- Cho, S.; May, G.; Tourkogiorgis, I.; Perez, R.; Lazaro, O.; de la Maza, B.; Kiritsis, D. A Hybrid Machine Learning Approach for Predictive Maintenance in Smart Factories of the Future. In *Advances in Production Management Systems. Smart Manufacturing for Industry 4.0, IFIP Advances in Information and Communication Technology*; Moon, I., Lee, G.M., Park, J., Kiritsis, D., von Cieminski, G., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 311–317. [\[CrossRef\]](#)
- Reina, A.; Cho, S.-J.; May, G.; Coscia, E.; Cassina, J.; Kiritsis, D. Maintenance Planning Support Tool Based on Condition Monitoring with Semantic Modeling of Systems. In *Enterprise Interoperability*; Zelm, M., Jaekel, F.-W., Doumeings, G., Wollschlaeger, M., Eds.; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2018; pp. 271–276. [\[CrossRef\]](#)
- Luo, W.; Hu, T.; Ye, Y.; Zhang, C.; Wei, Y. A hybrid predictive maintenance approach for CNC machine tool driven by Digital Twin. *Robot. Comput.-Integr. Manuf.* **2020**, *65*, 101974. [\[CrossRef\]](#)
- Montero Jimenez, J.J.; Schwartz, S.; Vingerhoeds, R.; Grabot, B.; Salaün, M. Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics. *J. Manuf. Syst.* **2020**, *56*, 539–557. [\[CrossRef\]](#)
- Cho, S.; May, G.; Kiritsis, D. A Semantic-driven Approach for Industry 4.0. In Proceedings of the IEEE 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), Santorini Island, Greece, 29–31 May 2019; pp. 347–354. [\[CrossRef\]](#)
- Cho, S.; Hildebrand-Ehrhardt, M.; May, G.; Kiritsis, D. Ontology for Strategies and Predictive Maintenance models. *IFAC-PapersOnLine* **2020**, *53*, 257–264. [\[CrossRef\]](#)
- Bloch, T.; Sacks, R. Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models. *Autom. Constr.* **2018**, *91*, 256–272. [\[CrossRef\]](#)
- Nilsson, J.; Sandin, F. Semantic Interoperability in Industry 4.0: Survey of Recent Developments and Outlook. In Proceedings of the 2018 IEEE 16th International Conference on Industrial Informatics (INDIN), Porto, Portugal, 18–20 July 2018; pp. 127–132. [\[CrossRef\]](#)
- Madugula, L. Applications of IoT in Manufacturing: Issues and Challenges. *J. Adv. Res. Embed. Syst.* **2021**, *8*, 3–7. [\[CrossRef\]](#)
- Dogan, A.; Birant, D. Machine learning and data mining in manufacturing. *Expert Syst. Appl.* **2021**, *166*, 114060. [\[CrossRef\]](#)
- Mami, M.N.; Grangel-González, I.; Graux, D.; Elezi, E.; Lösch, F. Semantic data integration for the SMT manufacturing process using SANSA stack. In *European Semantic Web Conference*; Springer: Cham, Switzerland, 2020; pp. 307–311.

26. Ahmadi, A.; Moradi, M.; Cherifi, C.; Cheutet, V.; Ouzrout, Y. Wireless connectivity of CPS for smart manufacturing: A survey. In Proceedings of the IEEE 2018 12th International Conference on Software, Knowledge, Information Management & Applications (SKIMA), Phnom Penh, Cambodia, 3–5 December 2018; pp. 1–8.
27. Wang, J.; Xu, C.; Zhang, J.; Zhong, R. Big data analytics for intelligent manufacturing systems: A review. *J. Manuf. Syst.* **2021**, *62*, 738–752. [\[CrossRef\]](#)
28. Kendall, E.F.; McGuinness, D.L. *Ontology Engineering*; Synthesis Lectures on the Semantic Web: Theory and Technology; Morgan & Claypool: San Rafael, CA, USA, 2019; pp. 1–136.
29. Tudorache, T. Ontology engineering: Current state, challenges, and future directions. *Semant. Web* **2020**, *11*, 125–138. [\[CrossRef\]](#)
30. Hashemi, P.; Khadivar, A.; Shamizanjani, M. Developing a domain ontology for knowledge management technologies. *Online Inf. Rev.* **2018**, *42*, 28–44. [\[CrossRef\]](#)
31. James, A.T.; Gandhi, O.; Deshmukh, S. Knowledge management of automobile system failures through development of failure knowledge ontology from maintenance experience. *J. Adv. Manag. Res.* **2017**, *14*, 425–445. [\[CrossRef\]](#)
32. Wan, S.; Li, D.; Gao, J.; Roy, R.; Tong, Y. Process and knowledge management in a collaborative maintenance planning system for high value machine tools. *Comput. Ind.* **2017**, *84*, 14–24. [\[CrossRef\]](#)
33. Liang, J.S. A process-based automotive troubleshooting service and knowledge management system in collaborative environment. *Robot. Comput.-Integr. Manuf.* **2020**, *61*, 101836. [\[CrossRef\]](#)
34. Polenghi, A.; Roda, I.; Macchi, M.; Pozzetti, A. Ontology-augmented Prognostics and Health Management for shopfloor-synchronised joint maintenance and production management decisions. *J. Ind. Inf. Integr.* **2021**, *27*, 100286. [\[CrossRef\]](#)
35. Ullah, F.; Habib, M.A.; Farhan, M.; Khalid, S.; Durrani, M.Y.; Jabbar, S. Semantic interoperability for big-data in heterogeneous IoT infrastructure for healthcare. *Sustain. Cities Soc.* **2017**, *34*, 90–96. [\[CrossRef\]](#)
36. Adamczyk, B.S.; Szejka, A.L.; Canciglieri, O. Knowledge-based expert system to support semantic interoperability in smart manufacturing. *Comput. Ind.* **2020**, *115*, 103161. [\[CrossRef\]](#)
37. Weichhart, G.; Panetto, H.; Molina, A. Interoperability in the cyber-physical manufacturing enterprise. *Annu. Rev. Control.* **2021**, *51*, 346–356. [\[CrossRef\]](#)
38. Margaria, T.; Steffen, B. (Eds.) The interoperability challenge: Building a model-driven digital thread platform for CPS. Lecture Notes in Computer Science. In *Leveraging Applications of Formal Methods, Verification and Validation, Proceedings of the 10th International Symposium on Leveraging Applications of Formal Methods, ISO/FA 2021, Rhodes, Greece, 17–29 October 2021*; Springer International Publishing: Cham, Switzerland, 2021; pp. 393–413. [\[CrossRef\]](#)
39. Canito, A.; Corchado, J.; Marreiros, G. Bridging the Gap between Domain Ontologies for Predictive Maintenance with Machine Learning. In *Trends and Applications in Information Systems and Technologies*; Springer International Publishing: Cham, Switzerland, 2021; pp. 533–543. [\[CrossRef\]](#)
40. Cao, Q.; Giustozzi, F.; Zanni-Merk, C.; Beuvron, F.D.B.D.; Reich, C. Smart Condition Monitoring for Industry 4.0 Manufacturing Processes: An Ontology-Based Approach. *Cybern. Syst.* **2019**, *50*, 82–96. [\[CrossRef\]](#)
41. Cao, Q.; Samet, A.; Zanni-Merk, C.; Beuvron, F.D.B.D.; Reich, C. Combining chronicle mining and semantics for predictive maintenance in manufacturing processes. *Semant. Web* **2020**, *11*, 927–948. [\[CrossRef\]](#)
42. Grüninger, M.; Fox, M.S. (Eds.) The role of competency questions in enterprise engineering. IFIP Advances in Information and Communication Technology. In *Benchmarking—Theory and Practice*; Springer: Boston, MA, USA, 1995; pp. 22–31. [\[CrossRef\]](#)
43. Noy, N.F.; McGuinness, D.L. Ontology Development 101: A Guide to Creating Your First Ontology. 2001. Available online: https://corais.org/sites/default/files/ontology_development_101_aguide_to_creating_your_first_ontology.pdf (accessed on 14 June 2022).
44. Arp, R.; Smith, B.; Spear, A.D. *Building Ontologies with Basic Formal Ontology*; Massachusetts Institute of Technology: Cambridge, MA, USA, 2015.
45. IOF PSS Ontology. Available online: <https://www.industrialontologies.org/product-service-system-wg/> (accessed on 14 June 2022).
46. Carmignani, G. An integrated structural framework to cost-based FMECA: The priority-cost FMECA. *Reliab. Eng. Syst. Saf.* **2009**, *94*, 861–871. [\[CrossRef\]](#)
47. International Electrotechnical Commission. *Analysis Techniques for System Reliability: Procedure for Failure Mode and Effects Analysis (FMEA)*; International Electrotechnical Commission: Geneva, Switzerland, 2006.
48. Z-BRE4K Project. Available online: <https://www.z-bre4k.eu/> (accessed on 14 June 2022).
49. ISO 14224; Petroleum, Petrochemical and Natural Gas Industries—Collection and Exchange of Reliability and Maintenance Data for Equipment, 2nd ed. ISO (International Organization for Standardization): Geneva, Switzerland, 2006.
50. Rousopoulou, V.; Vafeiadis, T.; Nizamis, A.; Iakovidis, I.; Samaras, L.; Kirtsoglou, A.; Georgiadis, K.; Ioannidis, D.; Tzovaras, D. Cognitive analytics platform with AI solutions for anomaly detection. *Comput. Ind.* **2022**, *134*, 103555. [\[CrossRef\]](#)
51. Cao, Q.; Beden, S.; Beckmann, A. A core reference ontology for steelmaking process knowledge modelling and information management. *Comput. Ind.* **2022**, *135*, 103574. [\[CrossRef\]](#)
52. Montero-Jiménez, J.J.; Vingerhoeds, R.; Grabot, B. Enhancing predictive maintenance architecture process by using ontology-enabled Case-Based Reasoning. In Proceedings of the 2021 IEEE International Symposium on Systems Engineering (ISSE), virtual, 13 September–13 October 2021; pp. 1–8.

-
53. Emmanouilidis, C.; Gregori, M.; Al-Shdifat, A. Context Ontology Development for Connected Maintenance Services. *IFAC-PapersOnLine* **2020**, *53*, 10923–10928. [[CrossRef](#)]
 54. Karray, M.H.; Ameri, F.; Hodkiewicz, M.; Louge, T. ROMAIN: Towards a BFO compliant reference ontology for industrial maintenance. *Appl. Ontol.* **2019**, *14*, 155–177. [[CrossRef](#)]