CRANFIELD UNIVERSITY

ALI M A AL-SHDIFAT

DEVELOPMENT OF A CONTEXT-AWARE INTERNET OF THINGS FRAMEWORK FOR REMOTE MONITORING SERVICES

SCHOOL OF AEROSPACE, TRANSPORT, AND MANUFACTURING

PhD
Academic Year: 2017 - 2020

Supervisor: Dr Christos Emmanouilidis
Associate Supervisor: Prof Andrew Starr
December 2020

CRANFIELD UNIVERSITY

SCHOOL OF AEROSPACE, TRANSPORT, AND MANUFACTURING

PhD

Academic Year 2017 - 2020

ALI M A AL-SHDIFAT

Development of a Context-Aware Internet of Things Framework for Remote Monitoring Services

Supervisor: Dr Christos Emmanouilidis
Associate Supervisor: Prof Andrew Starr
December 2020

This thesis is submitted in partial fulfilment of the requirements for the degree of PhD

LIST OF PUBLICATIONS

International Journal and Book Chapters

Al-Shdifat A, Emmanouilidis C, Khan M and Starr AG (2020) "Ontology-Based Context Modeling in Physical Asset Integrity Management". Frontiers in Computer Science. Sci. 2:578673. DOI: 10.3389/fcomp.2020.578673.

NB: Partial results from this publication have been used in the thesis chapters.

Al-Shdifat A., Emmanouilidis C., Starr A. (2020) "Context-Awareness in Internet of Things Enabled Monitoring Services". In: (eds) Engineering Assets and Public Infrastructures in the Age of Digitalization. Liyanage J., Amadi-Echendu J., Mathew J. Lecture Notes in Mechanical Engineering. Springer, Cham. https://doi.org/10.1007/978-3-030-48021-9_98.

NB: Partial results from this publication have been used in chapters 1 and 2.

International Conference Publications

Al-Shdifat A, Emmanouilidis C, Khan M and Starr AG (2020) "Ontology-based context resolution in internet of things enabled diagnostics". 4th IFAC Workshop on Advanced Maintenance Engineering, Services and Technologies at Cambridge, UK. (**Accepted**).

Al-shdifat, A., and Emmanouilidis, C. (2018) 'Development of a Context-aware framework for the Integration of Internet of Things and Cloud Computing for Remote Monitoring Services', Procedia Manufacturing, Elsevier, V16, pp. 31–38. https://doi.org/10.1016/j.promfg.2018.10.155.

NB: Partial results from this publication have been used in chapters 1 and 2

Emmanouilidis C, Gregori, M, **Al-Shdifat, A** (2020) "Context Information Management Ontology for Connected Maintenance Services". 21st IFAC World Congress, Berlin, Germany. (**Accepted**).

ABSTRACT

Asset management is concerned with the management practices necessary to maximise the value delivered by physical engineering assets. Internet of Things (IoT)-generated data are increasingly considered as an asset and the data asset value needs to be maximised too. However, asset-generated data in practice are often collected in non-actionable form. Moreover, IoT data create challenges for data management and processing. One way to handle challenges is to introduce context information management, wherein data and service delivery are determined through resolving the context of a service or data request.

This research was aimed at developing a context awareness framework and implementing it in an architecture integrating IoT with cloud computing for industrial monitoring services. The overall aim was achieved through a methodological investigation consisting of four phases: establish the research baseline, define experimentation materials and methods, framework design and development, as well as case study validation and expert judgment. The framework comprises three layers: the edge, context information management, and application. Moreover, a maintenance context ontology for the framework has developed focused on modelling failure analysis of mechanical components, so as to drive monitoring services adaptation. The developed context-awareness architecture is expressed business, usage, functional and implementation viewpoints to frame concerns of relevant stakeholders. The developed framework was validated through a case study and expert judgement that provided supporting evidence for its validity and applicability in industrial contexts.

The outcomes of the work can be used in other industrially-relevant application scenarios to drive maintenance service adaptation. Context adaptive services can help manufacturing companies in better managing the value of their assets, while ensuring that they continue to function properly over their lifecycle.

Keywords:

Internet of Things, Context Information Management, Maintenance Ontology, Cloud Computing, Remote Monitoring Services

ACKNOWLEDGEMENTS

I am thankful to Allah for all the blessings.

I would like to say a “Special Thanks” to certain people who, I believe, truly deserve them.

First and foremost, I would like to show my appreciation to my supervisor Dr Christos Emmanouilidis. I would not have been able to complete this thesis without his support, advice guidance and for helping me think outside of the box.

I will forever be indebted to Professor Andrew Starr who directed me through this whole process with his illuminating conversations. I am also thankful to Dr Muhammad Khan for providing constructive feedback on the project progression.

A sincere thanks goes to my loving parents for standing by my side and always encouraging me to push myself beyond my limits, and their long nights filled with prayers wishing me endless success in my future. This modest thesis is dedicated to them.

I would like to thank my brothers and sisters for their supportive thoughts and encouragement and a “special thanks” goes to my uncle Dr Ahmed and his family for motivating me to finish this journey.

I would like to thank my fellow colleagues for their supportive thoughts and encouragement.

TABLE OF CONTENTS

LIST OF PUBLICATIONS i
ABSTRACT i
ACKNOWLEDGEMENTS iii
LIST OF FIGURES vii
LIST OF TABLES x
LIST OF ABBREVIATIONS xi
1 INTRODUCTION 1
1.1 Overview 1
1.2 Aim and Objectives 6
1.3 Research Questions 6
1.4 Outline of the Thesis 7
2 LITERATURE REVIEW 11
2.1 Introduction 11
2.2 Overview of IoT and Cloud Computing 14
2.2.1 IoT Architectures 16
2.2.2 IoT Functionality 22
2.2.3 Middleware Support for IoT 25
2.2.4 Cloud Computing 27
2.2.5 IoT and Cloud Computing for Remote Monitoring Services 29
2.3 Context Information Management 32
2.3.1 Introduction 32
2.3.2 Context Types and Categorisation Schemes 34
2.3.3 Context-Awareness in IoT 36
2.3.4 Context Lifecycle Management 37
2.4 Context Information Sharing through IoT Platforms and Middleware 44
2.5 Ontologies in Predictive Maintenance and Asset Management 48
2.6 Chapter Summary and Research Gaps 52
2.6.1 Research Gaps 53
3 RESEARCH DESIGN AND METHODOLOGY 57
3.1 Introduction 57
3.2 Research Philosophical Approach 57
3.2.1 Research Philosophical Assumption 58
3.2.2 Philosophical Stance of this Research 58
3.3 Research Approach 59
3.4 Research Choice 60
3.5 Research Strategy 61
3.6 Time Horizon 61
3.7 Research Structure Diagram 62
3.7.1 Phase 1 – Establish the Research Baseline 62
3.7.2 Phase 2 –Framework Design and Development 63

3.7.3 Phase 3 – Experimentation Materials and Methods 64
3.7.4 Phase 4 – Framework Validation 66
3.8 Chapter Summary 67
4 ARCHITECTURE AND FRAMEWORK DEVELOPMENT 69
4.1 Introduction 69
4.2 Architecture Development 70
4.2.1 The Business Viewpoint 71
4.2.2 The Usage Viewpoint 73
4.2.3 The Functional Viewpoint 76
4.2.4 The Implementation Viewpoint 80
4.2.5 Summary 84
4.3 FRAMEWORK DESIGN AND DEVELOPMENT 85
4.3.1 Introduction 85
4.3.2 Overview of the context modelling and management framework 86
4.3.3 Three-Layered Context-Aware System Framework 93
4.4 Chapter Summary 98
5 MAINTENANCE ONTOLOGY DEVELOPMENT 101
5.1 Introduction 101
5.2 Failure Mode and Effect Analysis (FMEA) 105
5.3 Design of the FMEA-based ontology 108
5.3.1 Determine Scope 108
5.3.2 Consider Reuse 109
5.3.3 Enumerate Terms 109
5.3.4 Define Classes and Hierarchies 110
5.3.5 Define Properties and Constraints 111
5.3.6 Create Instances 113
5.4 Chapter Summary 114
6 CASE STUDY, RESULTS DISCUSSION, AND VALIDATION 117
6.1 Introduction 117
6.2 Case Study Results and Discussion 118
6.3 Ontology validation 125
6.3.1 Expert judgment 125
6.3.2 Ontology applicability, robustness, internal consistency, and effectiveness 129
6.4 Architecture Validation 135
6.4.1 Scenario 1 “Building Management System (BMS)” (Early Prototype) 136
6.4.2 Scenario 2 “Machine Monitoring System” 144
6.5 Chapter Summary 149
7 CONCLUSIONS AND FUTURE WORK 153
7.1 Contribution to the knowledge 153
7.1.1 Review of the Research Objectives 155

7.1.2 Review of the Research Questions 159
7.2 Research Conclusion 160
7.3 Research Limitations 161
7.4 Future Work 163
REFERENCES 165
APPENDICES 185

LIST OF FIGURES

Figure 1-1: IoT functionalities for industrial monitoring services 2

Figure 1-2: Outline of the thesis 9

Figure 2-1: Systematic review methodology 12

Figure 2-2: Map of the literature review 13

Figure 2-3: Evolution of the IoT 14

Figure 2-4: Common IoT application domains 15

Figure 2-5: Architectural models based on ISO/IEC/IEEE 42010 (source: AIOTI, 2018) 18

Figure 2-6: RAMI 4.0 reference architecture (source: ZVEI) 19

Figure 2-7: IIC and RAMI 4.0 working together to enable cross-domain interoperability (source: Shi-Wan et al., 2018) 21

Figure 2-8: Elements of the IoT (adapted from Burhan et al., 2018) 22

Figure 2-9: Mapping of the different IoT protocols layer (source: AIOTI, 2019) 23

Figure 2-10: Multi-access in Enterprise IoT (source: Cisco, 2019) 24

Figure 2-11: Types of cloud services and top benefits 28

Figure 2-12: Maintenance-related data-processing cycle (adapted from ISO, 2012) 30

Figure 2-13: A context-aware system in an IoT environment for monitoring services. 33

Figure 2-14: Comparison of context categorisation schemes 35

Figure 2-15: Comparison of context acquisition factors 39

Figure 2-16: Comparison of context modelling techniques (synthesised from (Strang and Linnhoff-Popien, 2004; Perera et al., 2014; Cabrera et al., 2017) 41

Figure 2-17: Comparison of context reasoning techniques (synthesised from Strang and Linnhoff-Popien, 2004; Perera et al., 2014; Cabrera et al., 2017) 43

Figure 3-1: Research philosophical assumptions 58

Figure 3-2: Types of research – Viewpoint of objectives (adapted from Kumar, 2005) 60

Figure 3-3: Research methodology 63

Figure 3-4: Early Prototype phase 64

Figure 3-5: Virtual Prototype phase 65

Figure 3-6: The research methodology selection (adapted from Saunders et al., 2018) 67

Figure 4-1: Architecture viewpoints (adapted from IIRA 2015; IIC, 2017) 70

Figure 4-2: Business viewpoint 72

Figure 4-3: Usage viewpoint for maintenance action recommendation service 75

Figure 4-4: Functional viewpoint 76

Figure 4-5: Structure of an operation domain 77

Figure 4-6: Information domain 78

Figure 4-7: Context sharing feature 79

Figure 4-8: Context-aware architecture for the integration of IoT and cloud computing for industrial monitoring services. 81

Figure 4-9: Summary of architecture viewpoints 84

Figure 4-10: Main IoT entities 86

Figure 4-11: Context taxonomy for remote monitoring services based on (Emmanouilidis et al., 2019; El Kadiri et al., 2016) 88

Figure 4-12: Upper-level ontology and characteristics 90

Figure 4-13: Representation of SWRL transitive rule based on (Nuñez and Borsato, 2018) 92

Figure 4-14: A context-aware framework for remote monitoring services 95

Figure 5-1: Visual representation of the system framework 103

Figure 5-2: The implementation of the proposed ontology 104

Figure 5-3: CAD Rendering of Drive System and Bearing Locations 107

Figure 5-4: The ontology development stages 108

Figure 5-5: Ontology classes 110

Figure 5-6: Hierarchy of level 1, 2 and 3 classes adapted from (Nuñez and Borsato, 2018) 111

Figure 5-7: RDF graph of component characteristics adapted from (Nuñez and Borsato, 2018) 113

Figure 6-1: Gear-misalignment machine 118

Figure 6-2: Context based adaptation interface 119

Figure 6-3: Query result to identify the main components of an asset..... 120

Figure 6-4: Query result to identify component functions 120

Figure 6-5: Characteristics of shaft key analysis 121

Figure 6-6: Misalignment faults matched to measurement parameters and techniques 122

Figure 6-7: Types of shaft misalignment (adapted from Khan et al., 2019) 123

Figure 6-8: Vibration velocity in RMS on context-based adaptation interface 132

Figure 6-9: SWRL rules for generating a warning 133

Figure 6-10: SWRL rules for generating potential cause 134

Figure 6-11: Implementation of high-level architecture..... 137

Figure 6-12: BMS system for monitoring a group of buildings 137

Figure 6-13: BMS for building A 138

Figure 6-14: Building A dashboard 139

Figure 6-15: Alarms system..... 140

Figure 6-16: Occupancy based on motion detection 141

Figure 6-17: Wind turbine rotated based on wind direction 142

Figure 6-18: DHT22 sensor connected to a Raspberry pi 2 (microcontroller). 143

Figure 6-19: Temperature and Humidity readings 144

Figure 6-20: Physical gearbox transmission test rig for emulating misalignment cases 145

Figure 6-21: Draft layout of the gear test rig presented on the ThingsBoard platform..... 146

Figure 6-22: Real-time temperature data 147

Figure 6-23: Time series of a vibration signal..... 148

LIST OF TABLES

Table 1-1: Relationships between the RQs and ROs. 7

Table 2-1: Characteristics of IoT cloud platforms (synthesised from Ismail et al., 2018; Hoffmann et al., 2019; Klaauw, 2019; Yu and Kim, 2019) 28

Table 2-2: Different context categorisation schemes..... 34

Table 2-3: Summary of context-aware system surveys..... 36

Table 2-4: Context sharing concerns..... 44

Table 2-5: Comparison of context-awareness features of existing approaches 46

Table 2-6: Ontologies used in maintenance and asset management..... 51

Table 2-7: Link gaps with research questions and sections 54

Table 5-1: A subset FMEA of Test Rig based on (del Castillo et al., 2020).... 106

Table 5-2: Object Properties adapted from (Nuñez and Borsato, 2018)..... 111

Table 5-3: Data Properties adapted from (Nuñez and Borsato, 2018) 112

Table 6-1: Query outcome for failure mode with highest DGN 124

Table 6-2: Information on the experts..... 126

Table 6-3: Level of knowledge scores 127

Table 6-4: Adjective rating scale – SUS score correlation..... 128

Table 6-5: Ontology usability evaluation..... 128

Table 6-6: Object properties using SWRL rules based on (Nuñez and Borsato, 2018) 130

Table 6-7: summarises how the elements of the proposed framework are validated. 150

Table 7-1: RQs and review informed from the research..... 159

LIST OF ABBREVIATIONS

AIOTI	Alliance for Internet of Things Innovation
API	Application Program Interface
CBR	Case-Based-Reasoning
CLA	Context Lifecycle Approaches
COBRA	Content Broker Architecture
CPS	Cyber-Physical System
ELA	Enterprise Lifecycle Approaches
EPC	Electronic Product Codes
ERM	Entity-Relationship Model
FMEA	Failure Mode and Effects Analysis
FMECA	Failure Modes, Effects and Criticality Analysis
HLA	High-Level Architecture
HTTP	Hypertext Transfer Protocol
IAAS	Infrastructure as a Service
IDC	International Data Corporation
IDS	Industrial Data Space
IIC	Industrial Internet Consortium
IIOT	Industrial Internet of Things
IIRA	Industrial Internet Reference Architecture
IOT	Internet of Things
IT	Information Technology
JSON	JavaScript Object Notation
KSOM	Kohonen Self Organizing Map
LAN	Local Area Network
M2M	Machine to Machine
MQTT	Message Queuing Telemetry Transport
ORM	Object Role Model
OWL	Web Ontology Language
PAAS	Platform as a Service
PAN	Personal Area Network
PHM	Prognostics and Health Management
PLM	Product Lifecycle Management
RAMI 4.0	Reference Architecture Model for Industry 4.0
RDF	Resource Description Framework

RFID	Radio Frequency Identification
SAAS	Software as a Service
SWRL	Semantic Web Rule Language
UCODE	Ubiquitous Codes
UML	Unified Modelling Language
WAN	Wide Area Network
XML	Extensible Markup Language

1 INTRODUCTION

1.1 Overview

In recent years, research into context management for Internet of Things (IoT) has received increased attention in academia, aimed to address the increasing complexity challenges of IoT-enabled data value chains (Perera et al., 2015). When considering IoT usage in industrial environments, the term 'Industrial Internet of Things' (IIoT), or simply 'Industrial Internet', is often employed, and is being considered synonymous to Industrie 4.0 (Jeschke et al., 2017). IoT is a set of enabling technologies that have the potential to provide ubiquitous connectivity to the Internet, transforming commonly-used objects into connected appliances. The underlying principle of IoT is the deployment of smart objects that have the capability of sensing the conditions in their surroundings, communicating and processing the obtained information, and then providing appropriate evaluations to the surroundings (Sisinni et al., 2018).

The rapid evolution of technologies associated with the deeper penetration of IoT in industry creates significant opportunities, but also introduces challenges for monitoring services. This is further fuelled by the accelerating shift to service-based business models, wherein service-level agreements must be ascertained, supported by adequate monitoring systems. IoT allows the physical world to be brought into the digital world, where physical things share information, and enables the coordination of decisions. Specifically, IoT brings together several functionalities, such as identification, sensing, communication, computation, services and semantics (Al-Fuqaha et al., 2015). Although IoT offers many benefits and solution enablers, substantial effort is required to manage and exploit the data generated by things. These functionalities can be used to connect a monitoring system with an end-user or another system, and can establish a remote overview of the observed system's state in order to prevent machinery performance degradation and reduce maintenance costs, as shown in **Figure 1-1**.

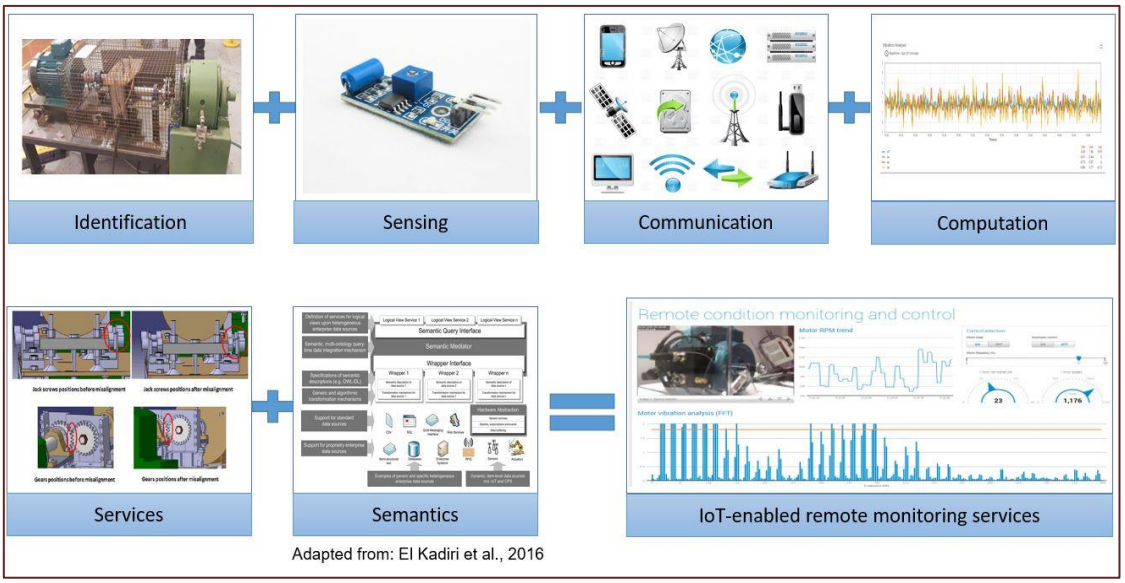


Figure 1-1: IoT functionalities for industrial monitoring services

IoT-generated data are increasingly considered to be an asset, and their data asset value needs to also be maximised. However, asset-generated data, in practice, are often collected in non-actionable forms. Such collected data may comprise a wide number of parameters, gathered over long periods of time, and possibly at significant scale. They may also fail to represent the range of possible scenarios of asset operation or the causal relationships between the monitored parameters, and so the size of the data collection, while adding to the complexity of the problem, might not necessarily allow direct data asset value exploitation. One way to handle data complexity is to introduce context information modelling and management, wherein data and service delivery are determined upon resolving the apparent context of a service or data request.

“Context is any information that can be used to characterize the situation of entities that are considered relevant to the interaction between a user and an application” (Dey et al., 2001). Context awareness is the ability of systems to give appropriate information or services to consumers utilising context information (Sezer et al., 2018). Systems with context awareness are employed in IoT environments for the purpose of sensing the operational environment and for delivering an appropriate response to both the user and application (Perera et al., 2014). Such systems are capable of analysing the data generated by IoT devices,

producing a high level of semantic organisation of the data and then converting it into context information. This information is subsequently utilised in determining an environment's status so as to drive appropriate responses. In general, the status of the environment is determined by a combination of circumstances, including users, applications, location or devices (Abowd et al., 1999), which constitute the context information.

As IoT technologies become more embedded in monitoring activities, there is a growing necessity to manage their context information in industrial environments. Numerous technologies can become a significant part of this, especially by making their connected devices work together. Cloud computing is particularly relevant, enabling the delivery of hosted services for instance, storage, networking, analytics and software development platforms over the internet (Zhou et al., 2013). This entails gathering, modelling, reasoning and disseminating context in order to efficiently manage the data generated by multiple devices, and to ensure that they can be effectively integrated into enterprise systems. Nonetheless, the data contributing to context information are often modelled or processed within the narrow scope of isolated subsystems, thus restricting interoperability. Moreover, even when similar systems for collecting context are applied in distinct settings, information is infrequently shared among them (Perera et al., 2014).

The ability to share context among different applications is a critical necessity for the IoT, making data shared between heterogeneous systems reusable in multiple applications (Ramachandran and Krishnamachari, 2019). Context information management has been recognised as a challenge for relevant research. Early on, Bernardos et al. (2008) developed a data fusion framework for context-awareness systems that included the following stages: (i) obtaining context; (ii) processing context; and (iii) reasoning and decision-making. Perttunen et al. (2009) surveyed popular context reasoning and representation techniques, providing an overview of the requirements for context representation, and arguing that such requirements were insufficiently covered in the literature regarding the interplay between efficiency, expressiveness, soundness and

completeness, with ontology-based approaches achieving improved scalability and reuse compared to other approaches. Ontology is defined as an explicit and shared conceptualization of a given domain (Dibley et al., 2012). It provides a vocabulary for representing knowledge about a domain which is often considered as a set of entities, relations, functions, and instances.

The findings of Bettini et al. (2010) supported this, although the scalability of online reasoning with a large number of entities was raised as a significant challenge. This is the case when dealing with data of significant complexity and scale, as typically encountered in IoT applications, making it important that the semantics of IoT data are captured by appropriate context modelling in order to gain valuable insights (Perera et al., 2014). The sheer complexity of such activities creates a need to narrow down the scope of processing, and ground it, if possible, in a sound domain. This is exactly where context information management can contribute. It plays a central role in determining what data need to be collected and how to process it. It also identifies what information and services are required to be presented to the consumer.

In the application domain of asset and maintenance management, context is relevant to the asset and its hierarchy, the user, the production or service business circumstances, as well as the overall system and operating-environment aspects (Emmanouilidis et al., 2019). The resolution of asset context is needed to analyse mechanical systems and logically connect measurements, observed behaviour and intended function with machinery operating conditions and faults. Thus, a knowledge construct can be used to resolve context resolution requests in order to drive maintenance services. Such resolution can be achieved by ontological reasoning based on semantic similarity, determined through ontological distance metrics or other appropriate methods (Teoh and Case, 2004). This bears relevance to similarity-based reasoning, such as that typically employed in Case-Based-Reasoning (CBR) systems, which have been employed in the past in the maintenance domain (Cândea et al., 2014). However, modelling and reasoning capabilities in ontologies go beyond CBR similarity.

For OWL2-based reasoning, the formulation of queries can be done via SPARQL queries in Resource Description Framework (RDF) documents. SPARQL is an RDF query language—that is, a semantic query language for databases—able to retrieve and manipulate data stored in RDF format. For Web Ontology Language (OWL2)-based reasoning, the formulation of queries can be done via SPARQL queries in RDF documents. OWL is a description logic based ontology language recommended by the World Wide Web Consortium (W3C) for use with the Semantic Web. Additionally, depending on the complexity of a given ontology model, the process of semantic matching can be served by using the Semantic Web Rule Language (SWRL). SWRL is a proposed language for the Semantic Web that can be used to express rules as well as logic. Overall, there is a need to further develop ontologically-based modelling, and an inference to drive maintenance services by extending currently-employed ontological concepts to include key additional and operational ones that are typically included in relevant standards, but less so in the relevant literature.

Therefore, context information management has largely dealt with the challenges of ubiquitous environments, as well as data heterogeneity and service scalability. Nonetheless, while substantial research efforts have been devoted to context information management in web-based, mobile and ubiquitous computing, including IoT-enabled computing, little attention has been given to translating these advances into tangible progress in industrial monitoring services (Al-shdifat and Emmanouilidis, 2018). Moreover, the most applicable context-modelling techniques that have been surveyed are ontology-based. However, the studied approaches lack some expressiveness concerning the knowledge representation for monitoring services in manufacturing environments. To address these needs, it is imperative that an effective context-aware framework is developed in order to enhance monitoring services in industrial environments as a means of addressing challenges related to information complexity, as well as to integrate data with domain knowledge in industrial monitoring applications. These observations led to the research aim and objectives described in the next section.

1.2 Aim and Objectives

The aim of the research is to develop a context awareness framework and implementing it in an architecture integrating IoT with cloud computing for industrial monitoring services. This is in order to address challenges related to information complexity, data heterogeneity, scalability, as well as integrating data with domain knowledge in industrial monitoring applications.

To achieve this aim, the following research objectives (ROs) were set for the study:

RO 1: To analyse the current practices of context lifecycle management, and identify the appropriate factors that are used in context acquisition, modelling, reasoning, and dissemination for IoT-enabled industrial monitoring services.

RO 2: To design and develop a framework and an architecture that introduces context-awareness to enhance remote monitoring services.

RO 3: To develop a maintenance context ontology for the framework focusing on modelling failure analysis of mechanical components.

RO 4: To apply the framework on a use case through an implementation architecture and validate it through experiments and expert judgement.

1.3 Research Questions

The research questions addressed by this study were developed in order to fulfil the research aim and objectives. The research questions were:

RQ 1: How can context be acquired, modelled, processed and disseminated for industrial monitoring services?

RQ 2: What is an appropriate framework to manage context awareness in a way that facilitates efficient condition monitoring?

RQ 3: How can the proposed framework, which integrates of IoT and cloud computing for industrial monitoring services, be validated?

Table 1-1 represents the extent to which the ROs contributed to answering the RQs (more ticks represent a stronger correlation).

Table 1-1: Relationships between the RQs and ROs.

	RO 1	RO 2	RO 3	RO 4
RQ1	√√√	√√	√	√
RQ2	√√	√√√	√	√√
RQ3	√	√	√√√	√√√

An outline of the thesis is given next.

1.4 Outline of the Thesis

This thesis is structured into seven chapters, as follows:

In **Chapter 1**, the motivation behind this research is discussed, with a description of the events that inspired the approach and the problem that offered the challenge. The RQs are stated here, alongside the respective specified ROs. An overview of the methodological approach adopted to address the ROs is given. Finally, the contribution of this work is summarised, and the document structure explained.

Chapter 2 provides an analysis of the findings of the literature review that were relevant to the targeted research area. Specifically, the literature review focuses on three main concepts–context Information management, IoT and cloud computing, and industrial monitoring services. Following the literature review, the problem statement is discussed, and the research gaps that indicated the need for this research are identified. This chapter fulfils RO 1 and RQ 1.

The different elements of the philosophical research approach are defined in **Chapter 3**, including philosophical assumptions, philosophical stances, research

approaches, research strategies, research choices and time horizons. The sources from which the data were gathered are also described in this chapter, which concludes with an illustration of the structure of the study.

In **Chapter 4**, a framework, and an architecture that introduces context awareness to enhance remote monitoring services are proposed. The framework applies context-aware computing to deliver solutions and address key challenges that IoT-enabled monitoring services need to handle, specifically how the context can be modelled, processed and disseminated for remote monitoring services, how this impacts the service discovery solution, and what an appropriate taxonomy of ontology is. This chapter fulfils RO 2 and RQ 2.

Chapter 5 presents a maintenance context ontology for the framework focused on failure analysis of mechanical components so as to drive monitoring services adaptation. The proposed ontology for the context resolution mechanism is relevant to the failure analysis of mechanical components, and the terminology and relationships between the concepts are structured on the basis of relevant standards, with a reliability-oriented knowledge grounding. This chapter fulfils RO 3 and RQ 2.

The focus of **Chapter 6** is on presenting a validation of the developed maintenance context ontology and framework. In this respect, a gear test rig was chosen as a case study to validate the applicability of the proposed framework and ontology. This test rig was designed to emulate complex cases of misalignment, relevant to manufacturing and aerospace engineering assets. This is followed by an analysis of the evaluations obtained from expert judgment of the applicability of the framework. This chapter fulfils RO 4 and RQ 3.

Chapter 7 provides a discussion of the work, and presents the overall critical findings of this research. Contributions to the literature are discussed and policy recommendations are provided. This chapter concludes with the limitations of this study and recommendations for the potential continuation of this research.

Figure 1-2 gives a diagrammatic representation of this thesis.

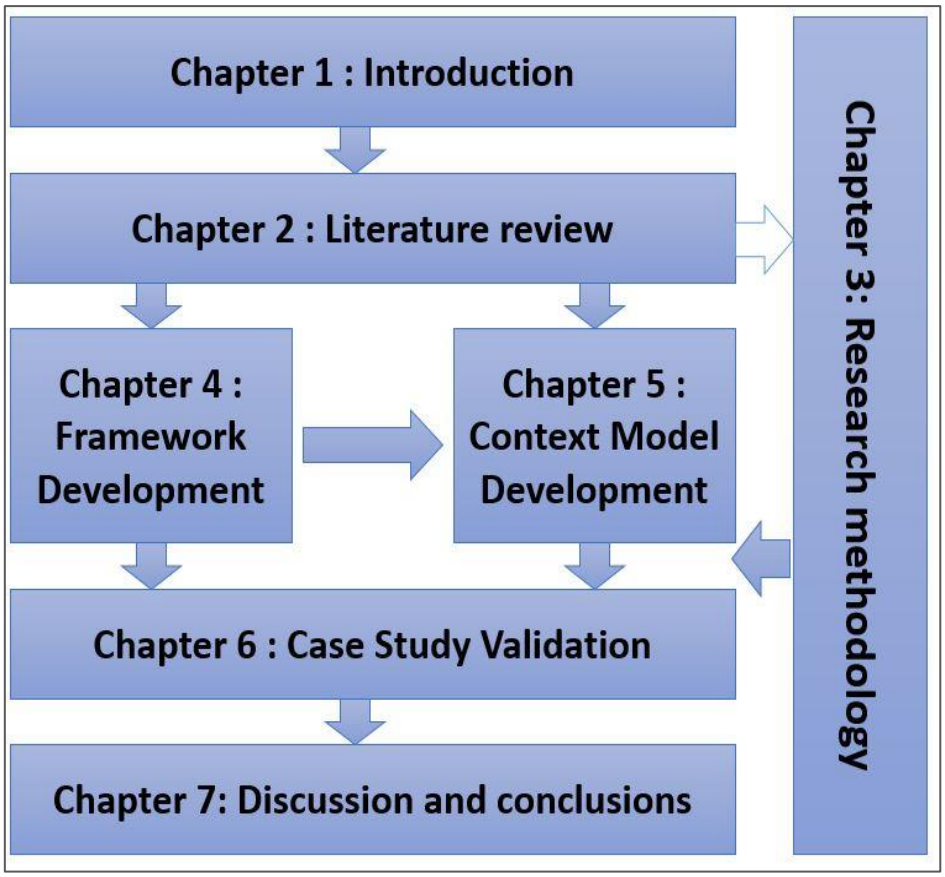


Figure 1-2: Outline of the thesis

2 LITERATURE REVIEW

2.1 Introduction

Industrial monitoring services are required in order to meet the very high demands on the availability and efficiency of industrial systems. The rapid evolution of technologies associated with deeper penetration of IoT in industry creates significant opportunities, but also introduces challenges for monitoring services. These are related to the entire data lifecycle, encompassing data acquisition, real-time data processing, transmission, storage, analysis and higher added-value service provision to users, with the adequate data management and governance required to be in place (Al-Shdifat et al., 2020). The sheer complexity of such activities, and the need to ground data processing on sound domain knowledge, emphasises the need for context information management to produce a semantic organisation of data so as to drive maintenance services adaptation. With this in mind, this chapter provides an analysis of the findings of a literature review that was relevant to the targeted research area. Specifically, the literature review focuses on three main concepts–context information management, IoT and cloud computing, and industrial monitoring services. This chapter starts with the systematic literature review for those three main concepts, then moves to an analysis of recent literature that addresses context information management in IoT.

A systematic literature review approach was applied to establish the research baseline and potential research gaps in the area. The main sources of information cited in this chapter were obtained from peer-reviewed journal articles in quality journals with acknowledged impact factors, books, reports and conferences proceedings in order to answer the RQs. The literature was surveyed by searching on a set of keywords in multiple databases (Scopus, Google Scholar, ABI, IEEE and Science Direct). The articles selected were mostly recent and directly related to one topic. An initial database search of selected keyword combinations revealed a total of 685 titles, as shown in **Figure 2-1**. After irrelevant subject areas were excluded, 340 titles in the relevant subject area remained. These results were narrowed down by applying another set of

keywords containing cloud computing, context lifecycle management and industrial monitoring services. This resulted in reducing the number of retained articles to 133 papers.

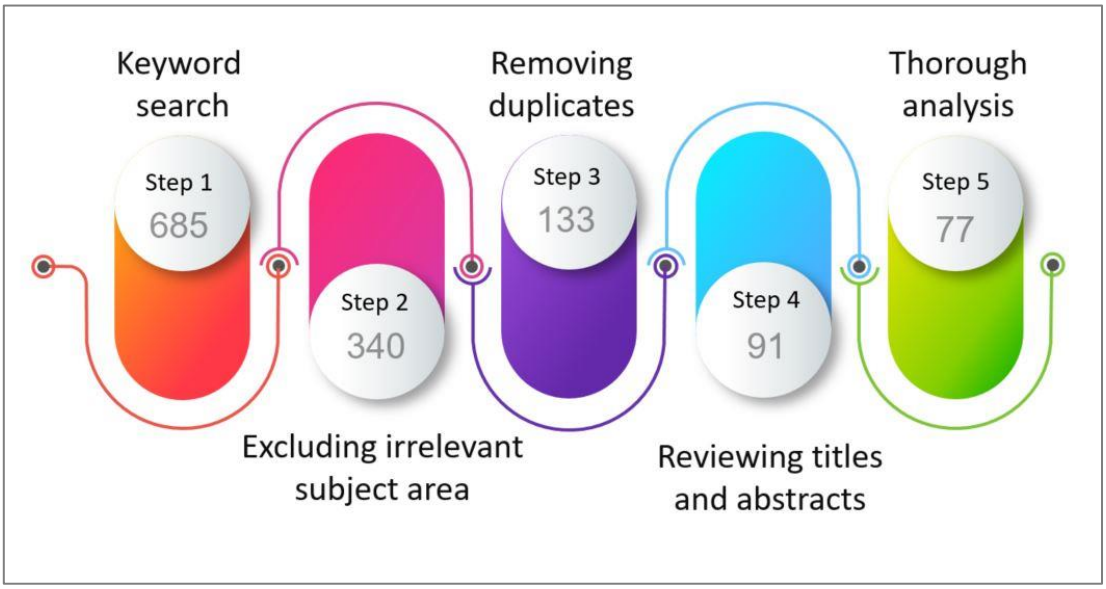


Figure 2-1: Systematic review methodology

Numerous studies had no close relation to the subject domain, although they did contain the keywords. Consequently, they were included in the search group. As shown in **Figure 2-2**, to ensure the relevance of all the articles, the titles, abstracts and keywords were carefully reviewed and qualified. Out of the 91 documents that remained subsequent to screening the titles and abstracts, only 77 were selected following a thorough analysis of them, with the others having no relevance to context information management in IoT, although they did lead to the identification of other relevant papers. Hence, they were not completely disregarded in this process, as they were read and cited where appropriate. The final 77 studies, published between 1999 and 2020, had a very strong relation with the subject matter.

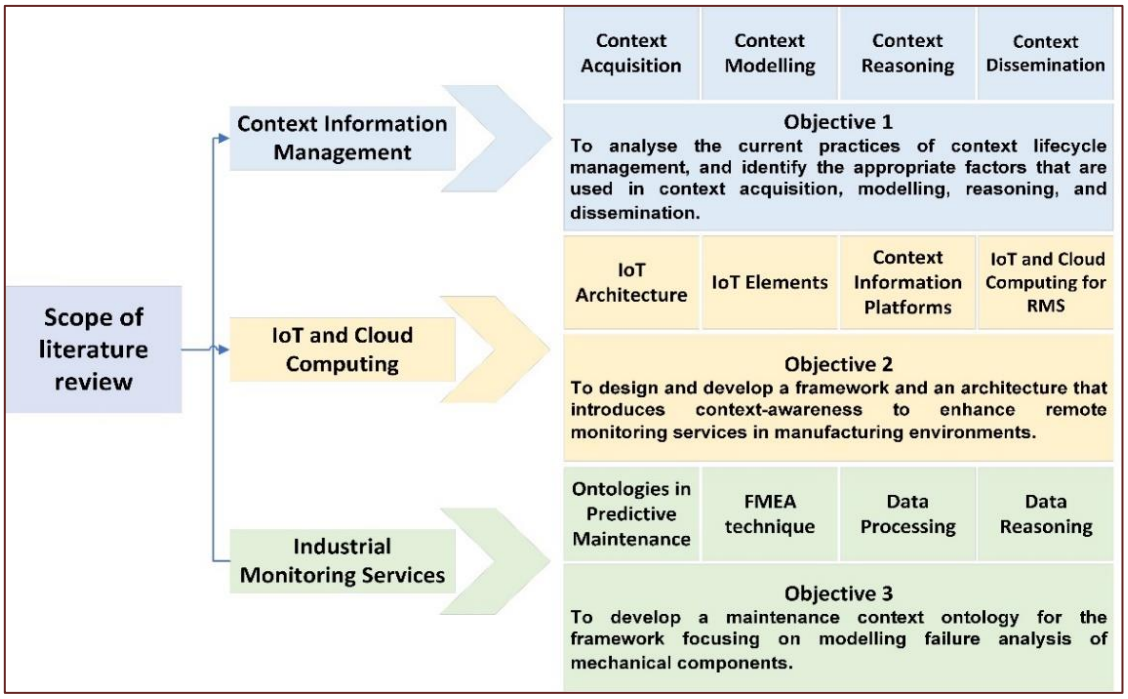


Figure 2-2: Map of the literature review

In order to establish the research baseline, as shown in **Figure 2-2**, a structured literature review in the area of context awareness in IoT and cloud computing for remote monitoring services was conducted. This chapter contains six core sections, starting with an overview of IoT, IoT architectures, IoT functionality, and finally middleware support for IoT (Section 2.2). This was important to understand the latest developments in this research area, and the technological trends that are extant today. This section also describes, in greater detail, the opportunities in cloud computing that provide the resources needed to store and analyse the vast amounts of data generated by IoT for remote monitoring services. Section 2.3 includes a review of context awareness, context types and categorisation schemes, and context lifecycle management as a way of determining the current practices in context lifecycle management, and identifying the appropriate factors used in context acquisition, modelling, reasoning and dissemination. The context information sharing through IoT platforms is then discussed in greater detail in order to establish how context awareness can support remote monitoring services (Section 2.4). Section 2.5 reviews ontologies in predictive maintenance and asset management. This is important in order to develop a maintenance context ontology for the framework focusing on modelling failure analysis of

mechanical components. The final section of the literature review (Section 2.6) is a summary of the lessons learned. The literature review led to the identification of research gaps.

2.2 Overview of IoT and Cloud Computing

Recent developments in the Fourth Industrial Revolution have led to a renewed interest in IoT for remote monitoring services in order to meet very high demands on the availability and efficiency of industrial systems (Uhlmann et al., 2015). The term 'IoT' has been credited to Kevin Ashton, one of the founders of the original Auto-ID Laboratory at MIT, who presented it in 1999. Typical applications of IoT technologies amalgamate the ability to identify, sense, compute, communicate and sometimes actuate for the purpose of monitoring and remotely controlling the environment (de Matos et al., 2020). The IoT has evolved broadly in five stages, as shown in **Figure 2-3**. Prior to the emergence of IoT, networking focused on connecting a few computer systems together, later moving to scale this up by creating the World Wide Web. Later, mobile devices and people were connected to the Internet via mobile and social networks. Eventually, IoT extended the Internet connectivity to internetwork, a wide range of physical entities (Perera et al., 2015).

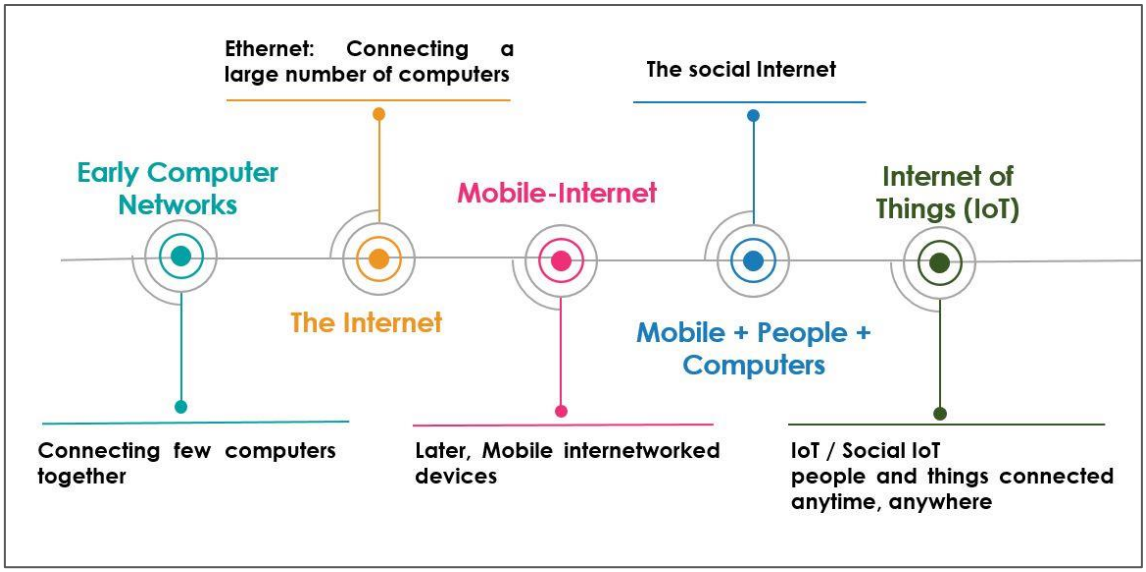


Figure 2-3: Evolution of the IoT

According to a Statista (2020) report, it has been predicted that the number of devices with Internet connectivity will exceed 50 billion by 2030. Such devices produce significant volumes of data, which are communicated through networks. Upon processing, such data enable better-informed decision-making and action-taking. Nonetheless, using data of significant scale for monitoring can be challenging for several reasons; for instance, processing time, data type, power, data size and storage. The sheer complexity of such activities creates a need to narrow down the scope of processing, grounding it, if possible, on sound domain knowledge. This is exactly where context information management can help. It plays a central role in determining what data needs to be collected and how it should be processed. It also identifies what information and services are required to be presented to a human or system actor.

IoT technologies are increasingly employed over a wide range of application domains, as shown in **Figure 2-4**. These include industry, healthcare, smart infrastructure, energy management and smart grids, retail and transportation, as well as many other areas that can transform our lives and societies for the better. A recent (2020) Statista report assumed that economic growth of IoT-based services was highly significant for businesses. According to projections, IoT expenditure is expected to undergo a compound annual growth rate of 13.6% for the period 2017–2025, exceeding $1.6 trillion by 2025 (Statista, 2020).

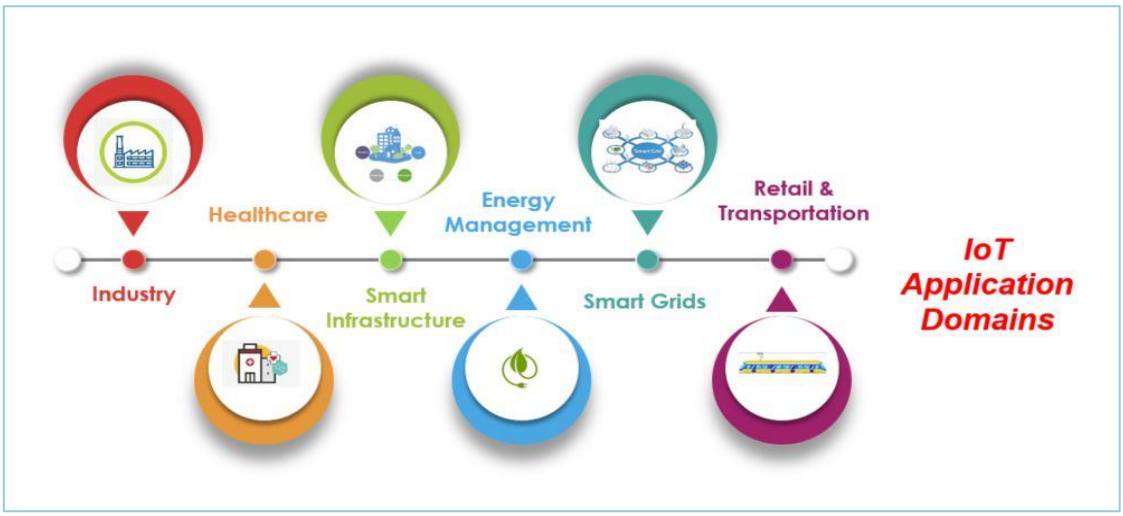


Figure 2-4: Common IoT application domains

One promising application area for IoT is that of industrial monitoring services. It can allow the connection of a monitoring system with an end-user or another system, and can establish a remote overview of the observed system's state in order to prevent machinery performance degradation, reduce maintenance costs, improve machine availability, and enhance process quality and safety. Furthermore, with global competition and technological progress, there have been growing demands by industry for more efficiency in monitoring the health status of manufacturing equipment in real-time. Remote monitoring services in the era of Industrie 4.0 are, however, facing some challenges, such as big data's 4Vs (volume, velocity, variety, veracity). While all these pose problems in conventional monitoring, they become even more challenging when integrating IoT and cloud computing to deliver advanced services to offer infrastructure availability and ubiquitous accessibility. Although IoT offers many benefits and solution enablers, substantial effort is required to manage and exploit the data generated by things services (Al-shdifat and Emmanouilidis, 2018).

Because IoT creates interconnections between a massive number of different things, facilitating the capability to sense, communicate and process data, it is extremely difficult to present an all-encompassing definition of the plethora of recent developments in this diverse a field. Nevertheless, various foundational aspects can be emphasised, namely IoT architecture, IoT functionality and middleware support for IoT. The next part of this chapter will discuss IoT architectures. This is important in order to achieve the research objective 2, which is to develop an architecture that introduces context-awareness to enhance remote monitoring services in manufacturing environments.

2.2.1 IoT Architectures

The proliferation of IoT has generated increased focus on architectural design and adaptive systems to facilitate the connectivity between heterogeneous IoT devices and IoT systems (Uviase and Kotonya, 2018). It is necessary for the blueprint of an IoT architecture to emphasise scalability, modularity and interoperability between heterogeneous devices that may utilise diverse technologies. Different types of IoT architectures have been proposed (e.g.

Industrial Internet Reference Architecture, www.iiconsortium.org; Reference Architecture Model for Industry [RAMI] 4.0, Zentralverband Elektrotechnik- und Elektronikindustrie [ZVEI]; Alliance for Internet of Things Innovation [AIOTI], https://aioti.eu/) to address several essential factors, such as sustainability, reliability, quality of service (QoS) and integrity. In general, the approach taken involves a description, with multiple layers arranged according to the different services provided at each layer, which are dependent on the chosen technologies in addition to the business demands and technical specifications. For instance, the International Telecommunication Union has defined an IoT architecture is comprised of five layers–application, middleware, networking, accessing, and sensing. Different works define IoT architectures in different layers, depending on the level of abstraction.

It was proposed by Atzori et al. (2010), Domingo (2012) and Jia et al. (2012) that IoT consists of three main layers–perception (or sensing), network and service (or application) layers. This three-layered architecture describes the main concept of IoT, but because research also focuses on the finer aspects of IoT (i.e. level of abstraction), this is not sufficient for IoT analysis. That is why some researchers have studied an additional layer that is also included in the current architecture of IoT–a middleware layer between the perception and application layers. In this regard, Liu et al. (2014) developed an IoT application framework that incorporates four layers– application, middleware, transport, and physical. In addition, in Xu et al. (2014), a framework consisting of four layers was obtained from the perspective of the functionalities provided, including sensing, networking, service and interface layers.

The AIOTI can be considered to be one of the main leaders in this area. A High-Level Architecture for IoT has been designed by AIOTI, which applies to large-scale AIOTI pilots. One of the key recommendations from the AIOTI is that any architecture should be represented utilising the ISO/IEC/IEEE 42010 standard. This standard specifies minimum requirements for the architecture of descriptions, frameworks, description languages and viewpoints, as shown in **Figure 2-5**.

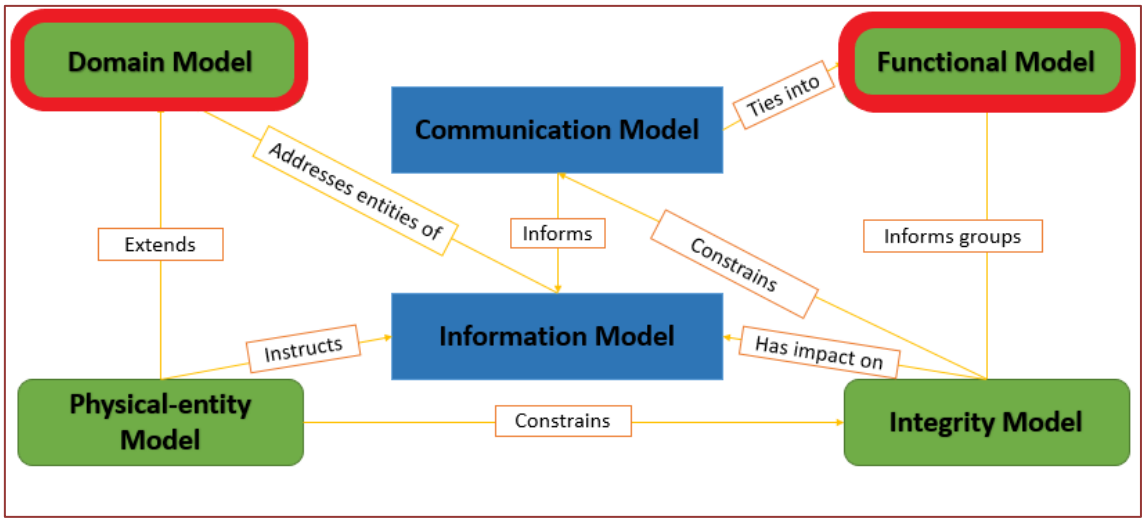


Figure 2-5: Architectural models based on ISO/IEC/IEEE 42010 (source: AIOTI, 2018)

Figure 2-5 distinctly illustrates how AIOTI WG3 concentrates upon functional and domain models. Domain models can be described broadly as the entities and relationships between them in the IoT domain (AIOTI, 2018). The functional model, from another perspective, defines both the interactions (interfaces) and functions in this domain.

According to the AIOTI (2018) report, the functional paradigm comprises four layers. These layers are perception, network, the IoT and application. The perception layer can be considered the interface between the informational world and the physical world layers. Several types of technology are utilised by this, including sensor technology, radio frequency identification (RFID), barcode technology and other types of sampling technology, the purpose of which is the completion of information data-gathering and the subsequent transfer of the data to the network layer. The data gathered from the perception layer can be transferred by the network or transmission layer to the IoT layer via various network technologies. Depending on the needs of the application entities, its main task is to provide short- and long-range connectivity, data reorientation between entities, and level control services, such as device triggering, QoS and location. Following that, the IoT layer creates specific functions for IoT, such as data sensing, storage, communication, services and semantics, and exposes the IoT

functions to App entities via the Application Program Interfaces (APIs). Finally, the application layer is the top layer of the functional model architecture. This can be considered to be the interface between the IoT and different types of users or systems and their specific needs as a means to achieve various smart applications of IoT. Consequently, the intelligent applications of IoT still need the support of information technology (IT), such as middleware, cloud computing and expert systems (Al-Fuqaha et al., 2015; Zhong et al., 2016).

In addition to the interaction between humans and machines, there is continuous communication between the machines. This generates an enormous and growing amount of data. Connectivity is one of the most essential features of the Fourth Industrial Revolution. It is a combination of the physical, natural and digital worlds. A RAMI 4.0 has been developed by the German Electrical and Electronic Manufacturers. RAMI 4.0 presents a general understanding of the relations existing among different individual components for the Industry 4.0 solutions landscape. RAMI 4.0 consists of three dimensions that describe the key features of Industry 4.0, as illustrated in **Figure 2-6**.

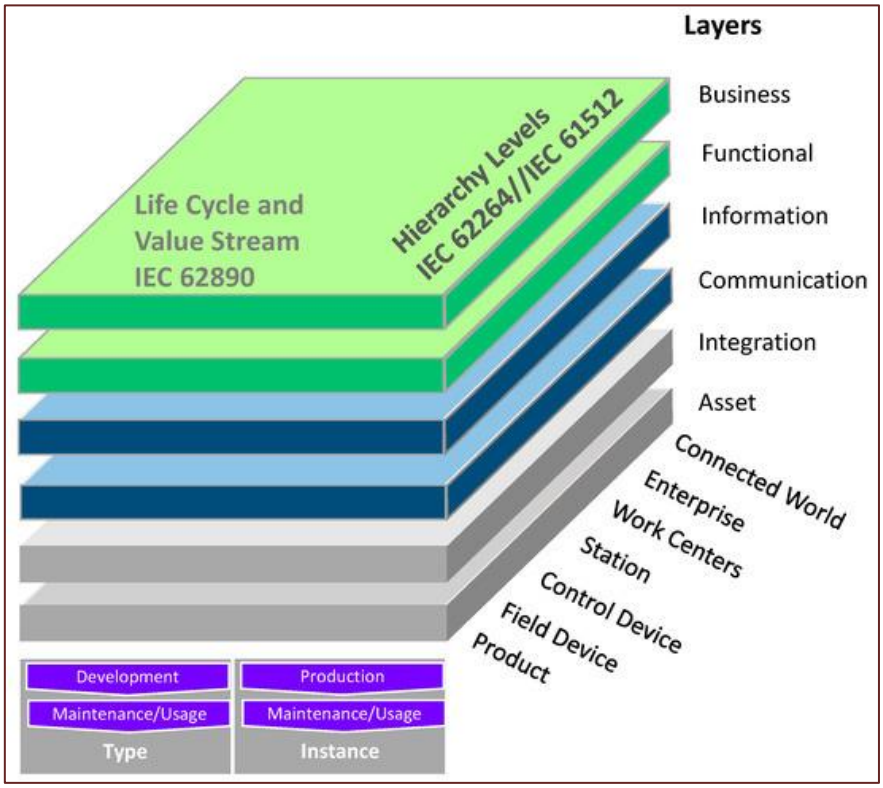


Figure 2-6: RAMI 4.0 reference architecture (source: ZVEI)

RAMI 4.0 is composed of several hierarchical levels and the lifecycle and value stream. On **Figure 2-6**, the vertical axis is made up of six layers that are used to describe the complex IT perspective as a composition of smaller manageable parts. The corresponding access layers (listed from bottom to top) are: asset, integration, communication, information, functional, and business (Contreras et al., 2017). The value chain and lifecycle are indicated by the left horizontal axis, which is divided into two sides (type and instance) based on a draft standard lifecycle management guideline known as IEC 62890. The component's function position in Industry 4.0 is indicated by the horizontal axis of the hierarchy levels. This adds the 'product' and 'field device' or the workpiece level at the bottom in order to expand the hierarchy levels. It also adds 'connected world', which, at the top, progresses beyond the separate factory boundaries (AIOTI, 2017; Wang et al., 2017). Therefore, Industry 4.0 is a specialisation within the IoT and services in the 'manufacturing environments' domain.

On the other hand, the Industrial Internet Consortium (IIC) is considered to be one of the leading global organisations, the function of which is to revolutionise the nature of businesses and society as a whole by promoting the implementation of the IIoT. This is achieved by facilitating reliable industrial internet systems in which secure connections are established between systems and devices, which are controlled in order to supply transformational results throughout different industrial sectors, including healthcare, transportation, energy, public domain infrastructure and manufacturing (Shi-Wan et al., 2018).

Both the IIC and RAMI 4.0 were designed based on the same objective of creating a convergence between the physical and digital domains, with a specific focus on converging IT and operational technology. There is obvious potential for generating mapping and alignment between the two in order to improve the comprehension of their complementary essence, as illustrated in **Figure 2-7**. This attempt at reconciling the architectures, in addition to the process of collaborating on testbeds and the design of infrastructures, will motivate collaboration throughout different industries and, ultimately, effect interoperability among systems. All of the above means that there is an increased potential to create

effective IIoT systems that provide improved business advantages and results (Shi-Wan et al., 2018).

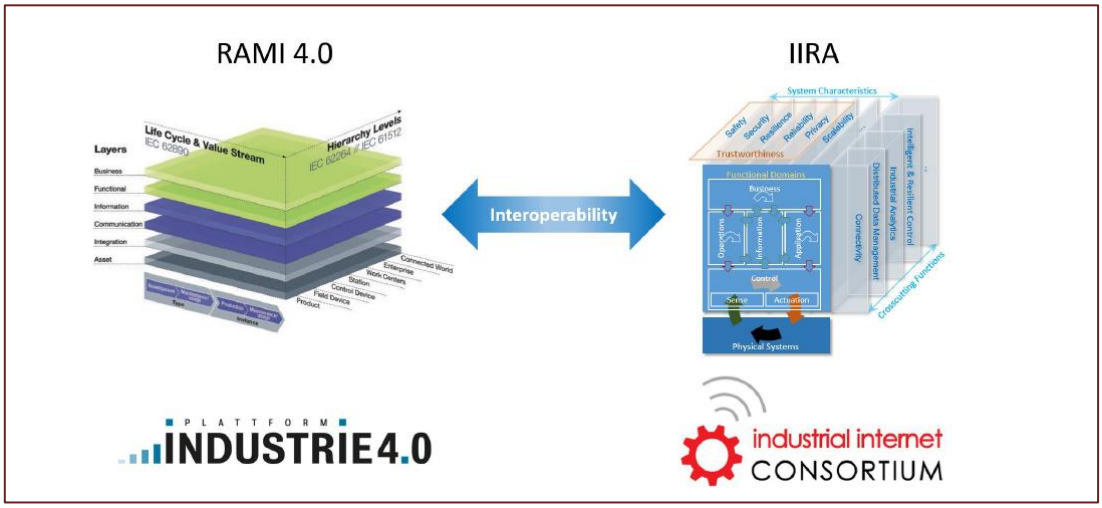


Figure 2-7: IIC and RAMI 4.0 working together to enable cross-domain interoperability (source: Shi-Wan et al., 2018)

RAMI 4.0 is concerned with developing products through the management of whole value chains in addition to the lifecycles of the products, whereas IIC is related to the construction, deployment and operation of large-scale connected systems. As manufacturing is also an example of an industry covered by IIC, both IIC and RAMI 4.0 can be applied in this sector. IIC emphasises widespread interoperability and applicability of its IIoT technical frameworks - with its reference architecture – throughout different industries. RAMI 4.0 extends to greater depths in terms of digitisation and the interconnectedness of manufacturing. For example, RAMI 4.0 incorporates various facets of manufacturing value chains throughout the entire product lifecycle, from the initial concept to the end of the product's life.

This section indicates that the transformation of society's demands towards technology-enabled services is a strong stimulus for improvements in industrial processes. This is further fuelled by the accelerating shift to service-based business models, wherein service-level agreements must be ascertained, supported by adequate monitoring systems. Context gathering, modelling, reasoning and dissemination are needed for the efficient handling of the vast

amounts of data produced by IoT-enabled devices, and their integration with other systems or industrial processes. In addition, the data collected by manufacturing information and communication technologies systems need to be introduced in a way that helps to improve machine availability and reduce maintenance costs. Consequently, there is a need for a flexible and more effective architecture to facilitate efficient monitoring systems that can be applied to a wider range of IoT-enabled monitoring applications. A thorough solution is required for the heterogeneity of IoT elements to make ubiquitous IoT services happen. The next section, therefore, moves on to discuss IoT functionalities, as well as their associated standards, technologies and realisations for the enhancement of monitoring systems.

2.2.2 IoT Functionality

The IoT brings together many functionalities, such as identification, sensing, communication, computation, services, and semantic information management (Al-Fuqaha et al., 2015; Burhan et al., 2018). **Figure 2-8** shows the elements needed to deliver the functionality of IoT.

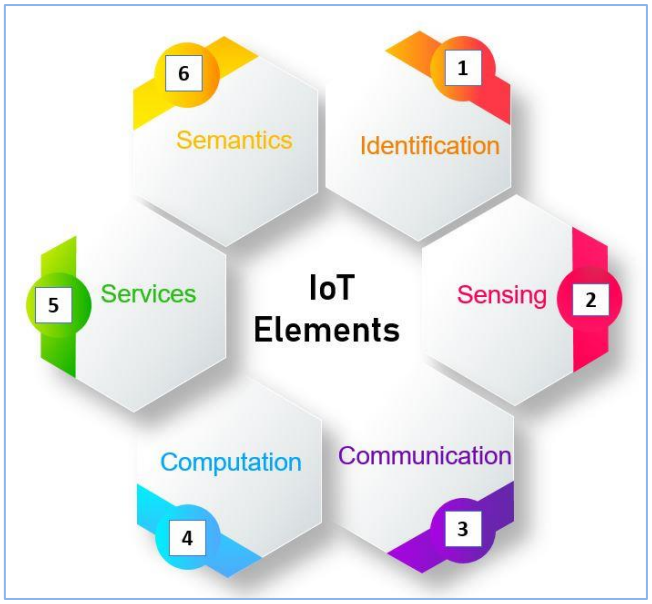


Figure 2-8: Elements of the IoT (adapted from Burhan et al., 2018)

1- **Identification** provides a specific identity for every object in a network. It consists of naming and addressing. Naming is where an object is named, whereas addressing gives a particular object a distinct address (Burhan et al., 2018). There are many identification methods used for IoT, such as ubiquitous codes (uCodes) and electronic product codes (EPC).

2- **Sensing** is extracting data from physical entities and sending it to other locations in order to analyse the collected data and to take specific actions based on the required services. Data are collected through sensors (i.e. temperature, pressure and humidity sensors and accelerometers) and are fed into an embedded computing device. The device is networked, and able to push the data to a cloud platform where they can be processed and visualised into relevant information for the user.

3- **Communication** technologies play a critical role in enabling the interconnectivity of things and the sharing of data between things and the Cloud. Several communication protocols currently exist for IoT, such as RFID, IEEE 802.15.4, Bluetooth and WiFi. **Figure 2-9** shows an example of the mapping of the different communication protocols provided for different IoT layers (AIOTI, 2019).

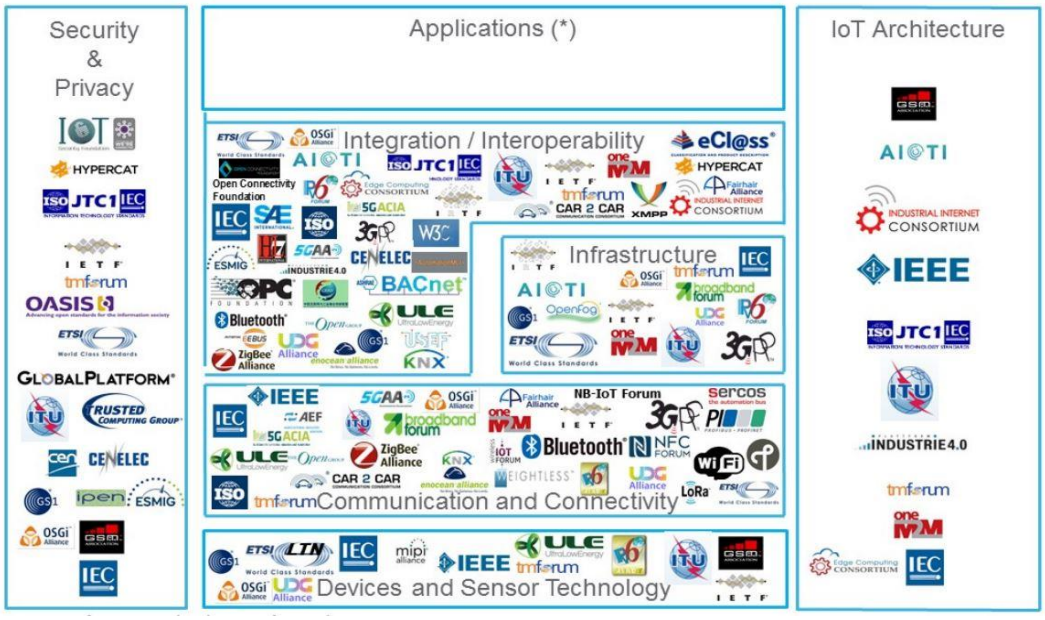


Figure 2-9: Mapping of the different IoT protocols layer (source: AIOTI, 2019)

The upcoming wave of connectivity technology (incorporating 5G) comprises what appears to be an unlimited diversity of cases and applications, including wearables and monitoring systems. Various technologies in IoT with the spectrum of Enterprise-level connectivity regarding some of their key characteristics (e.g. power consumption, bandwidth, type of interconnection), are shown in **Figure 2.10**.

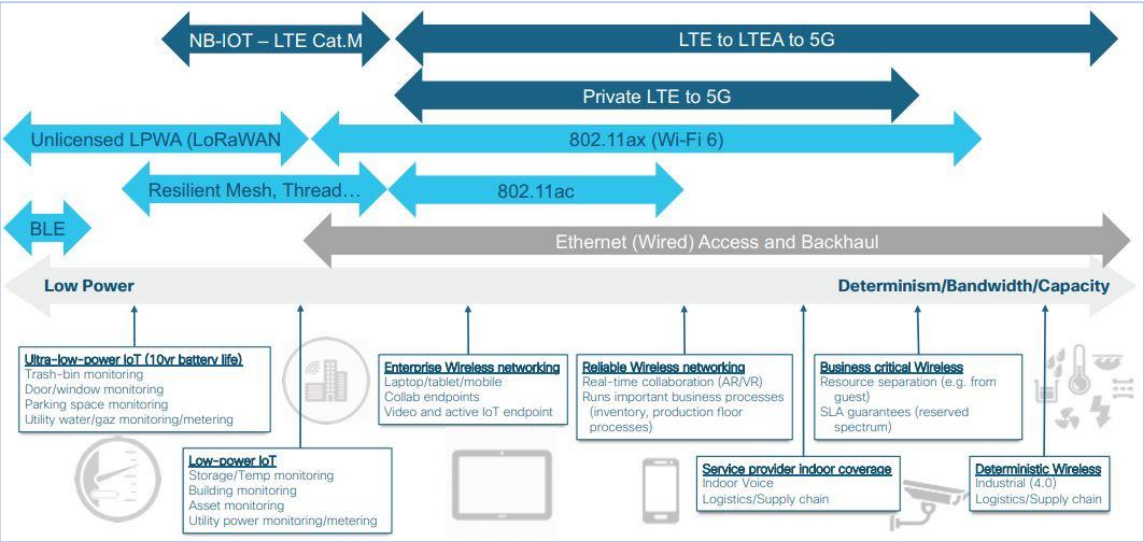


Figure 2-10: Multi-access in Enterprise IoT (source: Cisco, 2019)

4- **Computation:** IoT hardware and software applications are representative of the IoT's computational capability and its 'brain'. Computation allows a simple, highly customisable, cost-efficient way of acquiring data. Input signals originating from sensors can be processed by microcontrollers loaded with processing code that is uploaded into the memory and which sends output signals. This information can be presented in different forms, the most common being simple monitoring of the raw data on graphs to generate suggestions for possible interventions (Al-Fuqaha et al., 2015).
5- **Services:** IoT services can be divided according to multiple classifications, including collaborative-aware, identity-related, ubiquitous and information aggregation services (Friess and Vermesan, 2015). The most important and essential services that are used in other types of services are identity-related services (Al-Fuqaha et al., 2015). Any application that needs to carry real-world objects into the virtual world must first recognize them.

Raw sensory measurements that need to be processed and reported to the IoT application are gathered and summarized by Information Aggregation Services. Collaborative-Aware Services sit atop Information Aggregation Services and use the information gathered to make decisions and react appropriately. Ubiquitous Services, on the other hand, strive to provide Collaborative-Aware Services to everyone, wherever, whenever they are needed (Al-Fuqaha et al., 2015).

6- **Semantics** information processing offers solutions for managing the complex and often heterogeneous nature of data and knowledge from different entities, subsystems and users. The most commonly used semantic techniques are RDF, Extensible Markup Language (XML) and Web Ontology Language (OWL) (Barnaghi *et al.*, 2012).

Ubiquitous computing is the essence of IoT, which means integrating computing and communication into all objects around us. The interoperability of these heterogeneous devices requires well-defined standards. However, standardisation can be complicated due to the diverse needs of various devices and applications. A possible resolution to these heterogeneous applications is a middleware platform that can abstract an objects' details for applications. The following section will discuss middleware Support for IoT.

2.2.3 Middleware Support for IoT

Because the concept of IoT pivots on involving numerous devices that produce massive volumes of data, it is necessary to have software (i.e. IoT middleware) that can coordinate the interaction among IoT components. Middleware is considered a critical component of all IoT systems. It is essential because it facilitates an infrastructure that supports communication among heterogeneous devices, the abstraction of distinct applications, service discovery, the mobility of things, as well as privacy and security. Middleware can be described as a software layer that exists between the operating systems and the applications running on them. The importance of middleware was recognised early on in the computing literature, even before IoT had become a widely-adopted term. Middleware allows communication and data management for distributed

applications, and is essential for addressing the growing complexity of such systems (Issarny et al., 2002). Heterogeneity, interoperability, security and dependability are among the typical problems middleware provides solutions for in a reusable manner. The various features of IoT middleware solutions, such as device management, interoperation, platform portability and security, do not usually support context-awareness functionality, however; while some middleware solutions provided a level of context-awareness functionality, such requirements are insufficiently covered in the literature regarding device management, heterogeneity and interoperability (Atzori et al., 2010; Bandyopadhyay et al., 2011).

Although it is evident that IoT has significant potential and opportunities, the process of managing things to seamlessly integrate the physical and cyber worlds is still difficult (Raggett, 2015; Yao et al., 2015; Qin et al., 2016). An increasing number of IoT middleware and connectivity protocols have been developed. Nevertheless, a large number of these do not simplify the process of connecting IoT devices and then interpreting the subsequently collected data (Ngu et al., 2017). This problem is exacerbated by the fact that each type of IoT middleware supports various programming abstraction and architectures to access and connect to IoT devices.

The conclusions that can be drawn from this section are that certain significant functions are served by the middleware layer, including the gathering and filtering of data from hardware devices. IoT middleware is a mediator suite that hides the heterogeneity among the components, devices and technology of an IoT environment. Consequently, context gathering, modelling, reasoning and dissemination are needed for the efficient handling of the vast amounts of data produced by numerous devices and their efficient integration in Enterprise systems. Many technologies can be included as a significant part of this, especially by making connected devices work together. Cloud computing is particularly relevant, enabling the delivery of hosted services, such as storage, networking, analytics and software development platforms over the Internet. With this in mind, the next section moves on to describe, in greater detail, the

opportunities provided by cloud computing to provide the resources needed to store and analyse the vast amount of data generated in IoT for remote monitoring services.

2.2.4 Cloud Computing

A large volume of data is generated by IoT devices, which consequently places a significant strain on the Internet infrastructure. As a result, firms are required to determine solutions to minimise this pressure, as well as to find solutions to the problem of transferring significant volumes of data. It is necessary to store and process the large volume of heterogeneous data gathered by IoT devices, and the acquired insights must be retrieved for actuation or visualisation. However, it is rare that such tasks can be performed on IoT devices themselves, as their computing, storage, networking and energy resources are generally limited (Delicato et al., 2017). Thus, IoT must be supported by resources with greater power, with the most frequently used being cloud computing (Botta et al., 2016). Cloud computing has now become a standard IT that provides scalability in the provision of Enterprise applications and Software as a Service (SaaS).

Cloud computing represents a significant shift from the conventional way companies think about IT resources that can be utilised for remote monitoring services. It brings together several benefits, such as reliability, global scale, performance, speed and reduced costs. The term ‘cloud computing’ refers here to a model that allows users to access their applications, services and data at any time and from any location on the Internet as a result of the storage of information on the server supplied for cloud computing services and not on the user’s devices. Different kinds of service models exist according to the type of resources they deliver, as shown in **Figure 2-11**. The rapid provision and deployment of services, platforms and infrastructure are possible with less administrative work or interaction with the service provider (Mell and Grance, 2011).

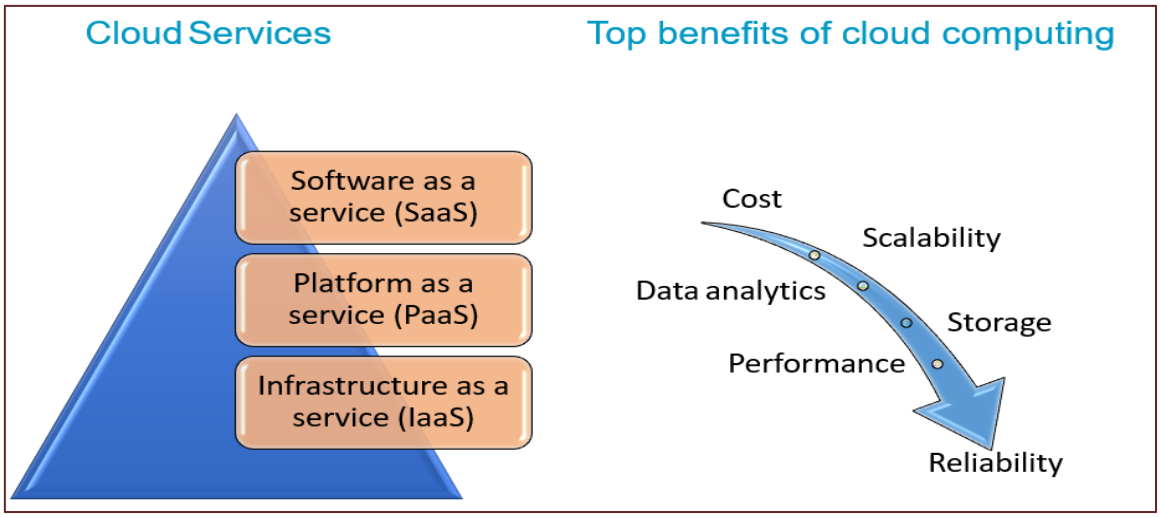


Figure 2-11: Types of cloud services and top benefits

Cloud computing architecture can be divided into three main categories. Firstly, Infrastructure as a Service provides infrastructure services and capabilities that deploy and run software in general, such as physical computers, virtual machines, networks, storage devices or a combination of these. Secondly, Platform as a Service plays a vital role in providing capabilities for application development and deployment in the cloud. Finally, SaaS provides the necessary applications and infrastructure services from the service provider, such as API.

Nowadays, there are several available IoT cloud platforms, both proprietary and open-source, that provide several operational benefits to industrial environments. IoT platforms are critical components of IoT architecture because they allow services and applications to be developed by end-users. IoT platforms vary in their capabilities and features, most being designed to solve major issues relating to the heterogeneity of objects and the cloud. The attributes regarded as being key to satisfying application developers' and users' requirements, according to the literature, are outlined in the platforms listed in **Table 2-1**.

Table 2-1: Characteristics of IoT cloud platforms (synthesised from Ismail et al., 2018; Hoffmann et al., 2019; Klaauw, 2019; Yu and Kim, 2019)

Platforms	Proprietary or Open-source	Functional Capabilities	Interoperability	Scalability	Security
Amazon	Proprietary	++	++	+++	+++

IBM	Proprietary	+++	++	++	+++
Microsoft	Open	+++	++	+++	+++
Google	Proprietary	++	++	+++	+++
FIWARE	Open	+	+++	+++	+
Cisco	Proprietary	+++	+	++	+++
ThingsBoard	Open	+++	+++	++	+
SiteWhere	Open	+	++	++	+
Huawei	Proprietary	+++	+	++	+

Although managing and controlling industrial IoT (IIoT) devices and data are important functions provided by IIoT platforms, various different platforms have been developed that are suitable for different use cases. IIoT platforms offer various combinations of capabilities, such as endpoint management and connectivity, the ability to capture, ingest and process IoT data, the visualisation and analysis of data, and the incorporation of IoT data into workflows and processes (Hoffmann et al., 2019). Furthermore, IIoT platforms provide a variety of different advantages, such as reduced costs, improvements in operations, production and security, as well as IoT data monetisation opportunities.

In summary, the spread of IIoT devices, standards and communications protocols is increasing the complexity of effectively managing IIoT networks. IIoT platforms are types of IIoT software that enable organisations to manage all the networked individuals, systems and objects within an IIoT ecosystem in a secure manner. Cloud computing delivers on-demand, convenient and scalable network access, which allows computing resources to be shared; in fact, this subsequently enables dynamic data from a variety of data sources to be integrated. The following section discusses the integration of IoT and cloud computing for industrial monitoring services.

2.2.5 IoT and Cloud Computing for Remote Monitoring Services

Although IoT and cloud have evolved separately, the literature has recognised that their integration has enabled them to jointly deliver many benefits. The effectively unlimited resources and capacity of the cloud has enabled IoT to gain

advantages. Cloud computing has a particular ability to enable the composition of web services and their utilisation by IoT applications. The fundamental concept behind IoT and cloud computing is that it makes regular tasks more efficient without having a detrimental impact on the quality of data that is being stored or transferred.

Product Lifecycle Management (PLM) systems are particularly benefitting from technologies that connect physical assets and products, processes, data, people and business systems (Keivanpour and Ait Kadi, 2019), exploiting product embedded-sensor and intelligence capabilities, including product or process Condition- Monitoring (CM) capabilities. Recent developments in IoT and cloud-computing technologies have led to a renewed interest in condition-monitoring systems. For the successful implementation of a condition-monitoring system, it is necessary to integrate various technologies that allow the flow of information from the data-collection stage through advisory generation-recommended actions. The process via which data is handled should comprise six distinct stages, as illustrated in **Figure 2-12**.

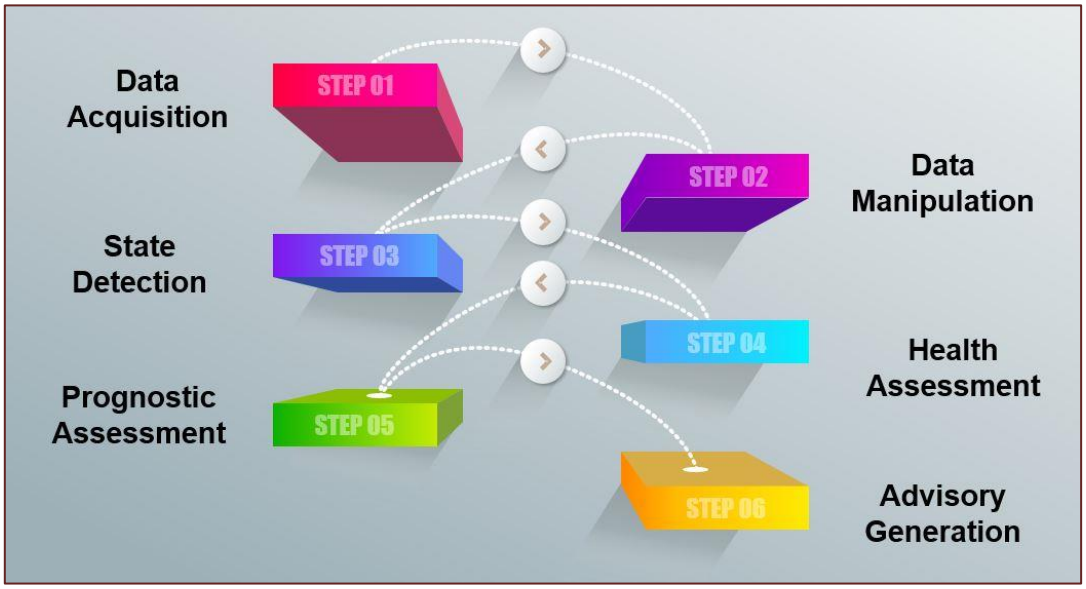


Figure 2-12: Maintenance-related data-processing cycle (adapted from ISO, 2012)

The first three stages are associated with technology, suggesting that they are dependent on a type of measurement. The data acquired from the transducer (sensor) is then digitised (defined as the data-acquisition stage) and processed

(data-manipulation stage), and then a comparison is made with standard baseline profiles, where mechanisms, such as thresholds for the detection of abnormalities, are also stipulated. The final three stages utilise the current state of the machine, and identify suitable maintenance actions on the basis of the expected state of the system or elements in the future. Initially, on the basis of historical data, and the diagnosis of any faults (fault-diagnosis stage), the existing situation can be analysed. Subsequently, future behaviour is forecast (prognostic-assessment stage), and then relevant maintenance actions are suggested (BSI ISO 13373-2:2016).

'Condition monitoring' refers to the process of acquiring and processing information and data that express the state of a machine over time, in order to provide meaningful real-time information to help reduce risk and lower rates (ISO, 2012). Through the four steps below, a remote monitoring programme, can remotely access asset-condition data and enable effective, efficient predictive maintenance (Peng, 2011).

1- Data acquisition is the process of gathering valuable data from things and converting that data into digital data.

2- Data processing involves producing meaningful information by converting digital data into real quantities of working machine conditions.

3- Decision-making not only focuses on the nature of machine failure, but also detects and identifies a machine fault. When failure happens, suitable actions could be considered automatically to control the operational status of the machine.

4- Remote communication is employed to transmit information, such as a machine's operational status and alarm conditions, over the network.

The integration of IoT and cloud computing can improve asset management processes through introducing more automation, real-time data analysis and smart decision-making. Moreover, such integration could reduce interruptions, improve uptime, remotely identify faults more quickly, and decrease time to repair by providing real-time data on the performance of assets that enhances their value to the business. Although several researchers have presented solutions

that focus on remote monitoring in the cloud, these solutions can only be used in specific scenarios, while being difficult to apply in other situations. IoT-driven and cloud-supported monitoring services are faced with some of the typical challenges of big data. To this end, it is necessary to introduce context awareness into this research in order to present contextual information about the involved objects, and to determine what information and services are required to be offered to the consumers, systems or software. Moreover, the context has a significant role in enabling the provision of adequate services to the users based on their surrounding environments. In addition, the context can help IoT services to adapt to dynamic environment changes and making right decisions. The following sections present a discussion on the basic concepts in the field of context-aware systems, including context, context awareness in IoT, and ontologies in maintenance and asset management.

2.3 Context Information Management

2.3.1 Introduction

The concept of context-aware systems was originally proposed by Schilit and Themier in 1994, who stated that “A system is context-aware, if it uses context to provide relevant information and/or services to the user”. Another early work defined context as “any information that can be used to characterize the situation of an entity, an entity is a person, place, or object that is considered relevant to the interaction between a user and an application” (Abowd et al., 1999). Abowd and Mynatt (2000) specified the basic elements required for analysing and understanding context, namely the five Ws (what, who, where, why and when). According to Byun and Cheverst (2004), a system is defined as being context aware if it is capable of extracting, interpreting and using context information, and its functionality can be adapted to the prevailing context of use. To clarify that, **Figure 2-13** illustrates an example of how context awareness applies in an IoT environment for monitoring services.

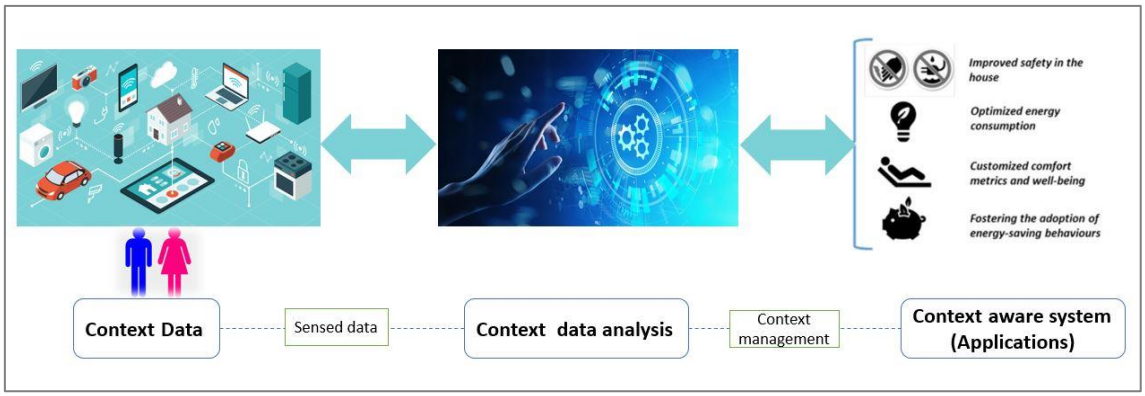


Figure 2-13: A context-aware system in an IoT environment for monitoring services.

Context awareness is the mechanism through which systems can adapt to the needs of a user by monitoring the context. Context includes environment, spatial, temporal, etc information that is used to infer the current activity. Collaboration among appliances is often ignored in traditional home design: each appliance performs its tasks and provides its services in isolation, with very minimal – or no – information exchange. This approach hinders the smart home paradigm's advancement of home automation and customisation of domestic services. In fact, in the smart home, domestic appliances should cooperate in a distributed and interconnected system, by acquiring context data and processing them in order to provide customised services. In this way, BMS scenarios can be used to create variations in the context, including rules of thumb for determining context-specific responses (control actions). For example, when ventilation is on and the difference between outdoor and indoor temperature is more than 5C, the indoor temperature approaches outdoor by 1C every 30 minutes. Otherwise, indoor temperature approaches outdoor temperature by 1C every 2 hours.

Highlighting the semantic added value of context, Sanchez et al. (2006) differentiated between raw data (data obtained directly from the source without being processed) and context information (raw data that has been processed). In the domain of asset and maintenance management, the early definition of context by Abowd et al. (1999) can be adopted and extended by specifying that context is relevant to the asset and its hierarchy, the user, the production or service

business circumstances, as well as the overall system and operating-environment aspects (Emmanouilidis et al., 2019).

2.3.2 Context Types and Categorisation Schemes

Despite the generally acknowledged definitions of what is regarded as being context awareness, a standard format and representation of the concept has not been established (Xu et al. 2014; Perera et al., 2015; de Matos et al., 2017; Tran et al., 2019; Hodkiewicz et al., 2020). Various researchers have determined different typologies of context. Abowd et al. (1999) differentiated between primary and secondary, in addition to conceptual and operational, context. It is possible to classify operational context further into sensed, static, profiled and derived, and, according to Chen and Kotz (2000), context can be categorised as passive and active, based on whether the context can be directly actioned with respect to the manner in which it is utilised in applications. Based on an operational categorisation viewpoint, Henricksen (2003) classified context into four levels—sensed, static, profiled and derived, whilst Liu et al. (2011) stated that context can be classified as user, physical or networking. Another study, published in the same year, by Yanwei et al. (2011) classified context into three levels—user, computing and things. **Table 2-2** provides a summary of the different approaches adopted in categorising context.

Table 2-2: Different context categorisation schemes

Survey	Year	Context Categorisation
Abowd et al.	1999	Who, Where, When What, and Why
Chen and Kotz	2000	User, Computing, Physical, and When
Henricksen	2003	Sensed, Static, Profiled and Derived
Van et al.	2005	Operational and Conceptual
Chong et al.	2007	Computing, Physical, Historical, and Sensor
Rizou et al.	2010	Observable and Non-Observable
Liu et al.	2011	User, Physical, and Networking
Emmanouilidis et al.	2013	User, Environment, System, Social, Service
Valverde et al.	2018	Location and Social

Various researchers have thus utilised different context categorisation schemes based on their different perspectives. An outline of the strengths and weaknesses of typical context classification approaches are depicted in **Figure 2-14**.

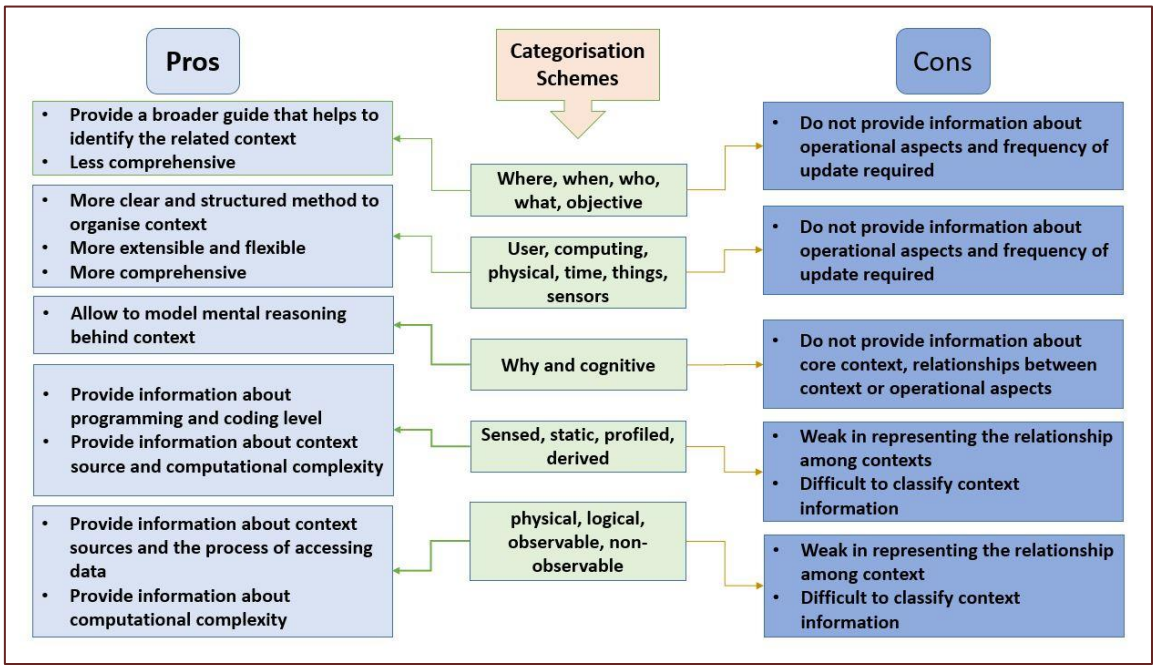


Figure 2-14: Comparison of context categorisation schemes

Context information management has largely dealt with the challenges of ubiquitous environments, as well as data heterogeneity and services scalability. Context management issues have increasingly progressed from dealing with context acquisition and modelling to context reasoning and dissemination. The early context information management literature targeted mobile computing and web-based information processing. IoT has expanded the range of applications having substantial need for context management, and this is reflected in the focus of relevant surveys. Several context-aware systems use context purely for decision-support/-making or direct distribution to the end-user. Nevertheless, certain systems allow context information to be shared with other interested actors or subsystems. This is defined as context information sharing, and represents one of the key challenges in the field of context awareness for IoT (Perera et al., 2015; Boavida et al., 2016).

2.3.3 Context-Awareness in IoT

Recent developments in IoT technologies have led to a renewed interest in context-aware computing. Context awareness plays an important role in defining what data needs to be gathered and how to process it, as well as in determining what information and services are required to be made available to a consumer of data or services (Perera et al., 2014; Sezer et al., 2018). Context management is generally considered to comprise context acquisition, modelling, reasoning and dissemination (Perera et al., 2014). **Table 2-3** summarises the surveys of context-aware systems conducted between 2010 and 2020.

Table 2-3: Summary of context-aware system surveys

Survey Title	Year	Contribution	Reference
"A survey of context modelling and reasoning techniques"	2010	Context modelling and reasoning in pervasive computing.	(Bettini et al., 2010)
"Context Aware Computing for The Internet of Things: A Survey"	2014	Comprehensive survey and analysis of context awareness for internet of things.	(Perera et al., 2014)
"Engineering context-aware systems and applications: A survey"	2016	Context-aware systems and applications in engineering.	(Alegre et al,. 2016)
"Internet of Things: A Review of Surveys Based on Context Aware Intelligent Services"	2016	A meta-survey of surveys on context awareness	(Gil et al., 2016)
"Context-Aware Computing, Learning and Big Data in Internet of Things: A Survey"	2018	Context awareness for IoT	(Sezer et al,. 2018)
"The MOM of context-aware systems: A survey"	2019	Comparison of several context-aware systems	(Pradeep and Krishnamoorthy, 2019)
"Context information sharing for the Internet of Things: A survey"	2020	Context sharing platforms- challenges and issues.	(de Matos et al., 2020)

According to Perera et al. (2015), the steps required for a system to deliver context information are acquisition, modelling, reasoning and distribution, which form the context lifecycle when combined. In the acquisition step, the raw data are collected from sensors, databases or the surrounding environment. In the modelling stage, the data is brought into a particular representation so it can be converted into input for the reasoning stage. Various approaches have been

described in the current literature for the modelling process, including key-value pairs, ontology and markup schemes (Bettini et al., 2010; Snidaro et al., 2015). The semantic processing stage in the context lifecycle is the reasoning process. Different methods can be used to infer context, such as supervised/unsupervised learning, rules, ontologies, probabilistic approaches, data aggregation and fusion mechanisms (Perera et al., 2015). Hence, the context awareness of a system is determined by its ability to utilise the context acquired via the context lifecycle in order to deliver beneficial information/services to the users (Abowd et al., 1999). Therefore, in order to design an appropriate framework to efficiently manage context for IoT-enabled monitoring services, further analysis is needed. In order to do that, an outline of how the lifecycle of context information can be managed must first be introduced.

2.3.4 Context Lifecycle Management

Context lifecycle management refers to how data is gathered, modelled and processed, and how knowledge is deduced from the obtained data (Sezer et al., 2018). It presents how data moves from stage to stage in software systems. It also demonstrates where the data comes from and where it is consumed (Jih et al., 2009). Hynes et al. (2009) suggested two categories of data lifecycle. The first is context lifecycle approaches (CLAs), which focus on context management. The second category is Enterprise lifecycle approaches, which deal with context and CLAs. There are also three stages of the typical context lifecycle—context acquisition, then information processing and, ultimately, reasoning and decision-making (Bernardos et al., 2008).

Context lifecycle management generally consists of four steps—context acquisition, modelling, reasoning and dissemination. The first stage involves obtaining the context from multiple sources, and compiling it to present more accurate data. Then, the obtained data must be represented and modelled in an appropriate form. After that, it is essential to obtain high-level context information based on the acquired contextually-relevant data through the processing of modelled data. The last step is distributing the context information to the interested parties (Perera et al., 2014).

2.3.4.1 Context Acquisition

Context acquisition involves acquiring and bringing together data from physical objects in different ways. In context acquisition, there are five factors that must be considered when context-aware data-processing in IoT model undergoes improvements. Each technique is explained below, based on sensor type, responsibility, acquisition process, frequency and source (Aguilar et al., 2018).

- **Based on responsibility**: Pietschmann et al. (2008) discussed context acquisition as being achieved using the pull and push methods.

  Pull can be defined as the programme component whose main function is to obtain sensor data from things regularly or immediately.

  Push is responsible for sending collected data from the physical or virtual sensor to the software component whose main function is to obtain sensor data from things regularly or immediately. Although the requirements and surrounding contexts change, the sensors are requested to programme and adapt to such changes.
- **Based on frequency**: Two types of events—instant and interval—can be the cause of context.

  Instant events occur immediately, and do not take long to happen.

  Interval events last for a period of time, and sensor data needs to be obtained regularly.
- **Based on Source**: Chen et al. (2004) argued that context can be acquired based on the source via two methods–directly from sensor hardware or through a middleware infrastructure. A positive aspect to this categorisation is the ability for direct communication with the sensors and fewer required resources. However, a negative aspect includes having less control over the sensor configuration.
- **Based on Sensor Types**: Acquiring context can be done by different types of sensors (Indulska and Sutton, 2003).

  Physical sensor are the most widespread and most used in our daily lives. They can be used to obtain data such as humidity, temperature and vibration.

Virtual sensor can be used to collect data that is difficult to physically measure because it does not generate data by itself, such as social networking, emails and calendars.

Logical sensors can be employed when data is required from a physical sensor and a virtual sensor in order to provide a meaningful information format.

- **Based on Acquisition Process**: Perera et al. (2014) argued that context can be acquired based on three techniques:

  Sensed: Sensors can be used to obtain data; for example, vibration can be retrieved through a vibration sensor. Also, some data can be obtained from databases.

  Derived: Data collected from sensors can be derived by calculations to produce other information.

  Manually provided: Some instructions and conditions are defined manually; for example, a user may want to be alerted when a temperature or vibration rises higher than normal. Researchers have used different techniques for context acquisition, each with their own strengths and weaknesses, as indicated in **Figure 2-15**.

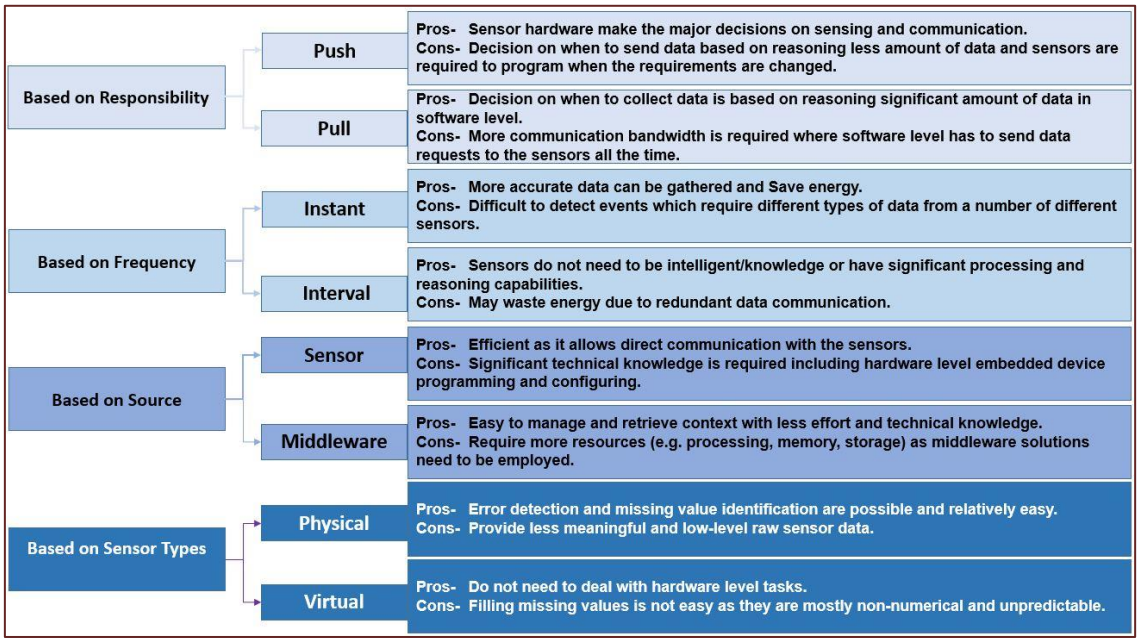


Figure 2-15: Comparison of context acquisition factors

2.3.4.2 Context Modelling

Context modelling is also generally referred to as the representation and formalisation of the context through certain modelling approaches (Cabrera et al., 2017). Perera et al. (2014), Veiga et al. (2017) and Cabrera et al. (2017) argued that, in order for context information to be processed into a format that could meet IoT requirements (i.e. expressiveness, reuse, extension and interoperability, among other things), they should go through a context modelling phase, which would present and give meaning to the collected context data. Context models come in two different types—static and dynamic—as described by Yanwei et al. (2011). Context modelling techniques have been examined by Chen and Kotz (2000), Strang and Linnhoff-Popien (2004), and Perera et al. (2014) and include ontology-based, key-value, logic-based, markup scheme and graphical methods.

- **Key-Value Modelling**: involves context information being modelled as key-value pairs in several formats that provide user-friendliness, flexibility and simplicity. Modelling limited amounts of data, such as application configurations, is among its other uses. Whilst it would be difficult to represent relationships using this model, it is useful for smaller amounts of data because it is flexible and easy to manage.
- **Markup Scheme Modelling**: Tags, which store context, are used to model data. Popular markup techniques, such as XML, have access to sophisticated validation tools. Whilst it can get complicated when many levels of information are involved, this technique can be used as an intermediate data organisation format that allows efficient data retrieval, which is why it is considered to be a step up from key-value modelling.
- **Graphical Modelling** exceeds both key-value and markup scheme modelling because it enables relationships to be obtained from the context model. Moreover, while that is easy, capturing queries can be complicated. According to Li et al. (2015), the most widely used model examples are the Unified Modelling Language, the entity-relationship model and the object-role model.
- **Logic-Based Modelling**: High-level context can be generated using low-level context, which is defined as facts, expressions and rules. When it comes to adding, updating or removing data, this model can be quite flexible.

❖ **Ontology-Based Modelling**: Ontology is defined as an explicit and shared conceptualization of a given domain (Dibley et al., 2012). It provides a vocabulary for representing knowledge about a domain which is often considered as a set of entities, relations, functions, and instances. Cabrera et al. (2017) pointed out that some of the uses of this model were in describing the relationships of the context and entities in the environment, and providing reasoning capabilities and data structure for data sources. Semantic technologies are used to organise the context into ontologies.

Several researchers have utilised different techniques for context modelling. These techniques have their own strengths and weaknesses, as shown in **Figure 2-16**.

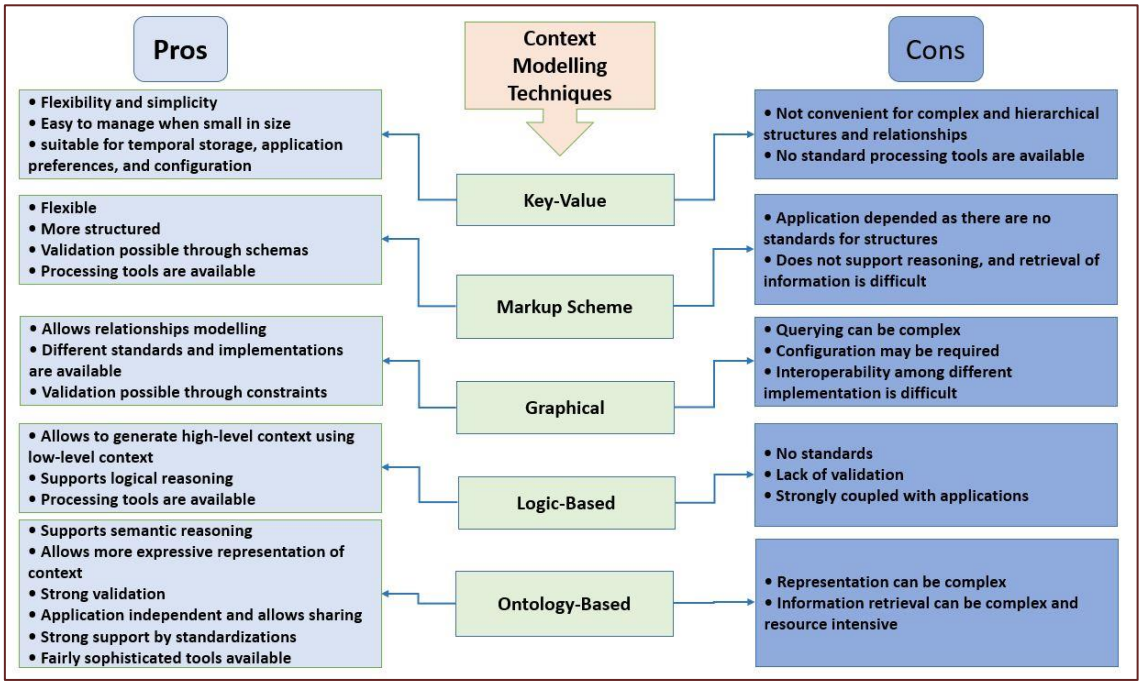


Figure 2-16: Comparison of context modelling techniques (synthesised from (Strang and Linnhoff-Popien, 2004; Perera et al., 2014; Cabrera et al., 2017)

The contextual information that is represented by a context model is easily stored and accessed by the mechanism on which the said paradigm depends. Each piece of contextual information becomes relevant only when it is coupled with another piece of information; when isolated, it makes no sense. Therefore, since contextual information is interdependent, a network is the best means of

representing it, rather than a sequential format, thereby allowing the semantics to be preserved. Consequently, the ontological method is one of the most popular methods of storing and organising contextual information because it is effective at storing densely interrelated and complicated pieces of information, which is a principal requirement for context-aware systems. It may be illustrated as a dense graph structure that represents the pieces of information as well as the ways in which they are connected.

2.3.4.3 Context Reasoning

Context reasoning can be defined as a process that contributes significantly to the collection of new knowledge based on acquired contextually-relevant data (Bikakis et al., 2008). Typical context reasoning techniques include rule-based approaches, supervised learning, fuzzy logic, unsupervised learning and ontology-based methods (Bikakis et al., 2008; Perttunen et al., 2009; Bettini et al., 2010):

- **Supervised Learning**: Gathering training examples is the first set of processes in this category. Classifying them according to the expected results comes second. The final step involves their application to all available data so that the anticipated results may be generated. It can describe decision tree, Bayesian and artificial neural networks as monitored learning methods that can be applied to model complicated paradigms and patterns between inputs and outputs.
- **Unsupervised Learning**: In this technique, the process of clustering is applied to derive significant results from data that has not been labelled. The k-nearest neighbours, Kohonen self-organising map, noise and outlier detection, and support-vector machines techniques are utilised for context reasoning. A clustering technique can be employed to resolve low-level, simple actions and operations (location and positioning).
- **Rules**: A reasoning technique can be obtained using an if-then format, which is easy to define, but can only be defined manually. A low-level context is used to create high-level context information. No other method rivals this in popularity; it is massively employed, coupled with ontological reasoning.

- ❖ **Fuzzy Logic** has certain differences to conventional logic. In conventional logic, everything is denoted by either 1 or 0. Conversely, in fuzzy logic, partial truth is also acceptable. Consequently, it is more admissible to the represented reality to use fuzzy logic than conventional logic (e.g. speed could be relatively fast, extremely slow, etc.). The fuzzy logic reason method is not used alone, however, being commonly applied in combination with ontological, probabilistic and rule-based reasoning approaches.
- **Ontology-based Reasoning**: Because ontology is based on description logic, it allows complex reasoning and complex representation. With this technique, data needs to be modelled in a compatible format, and this is one of its disadvantages. RDF(S), OWL and RDFS are common representations of semantic web languages for implementing ontology-based reasoning.

Researchers have utilised several different techniques for context reasoning, each with their own strengths and weaknesses, as indicated in **Figure 2-17**.

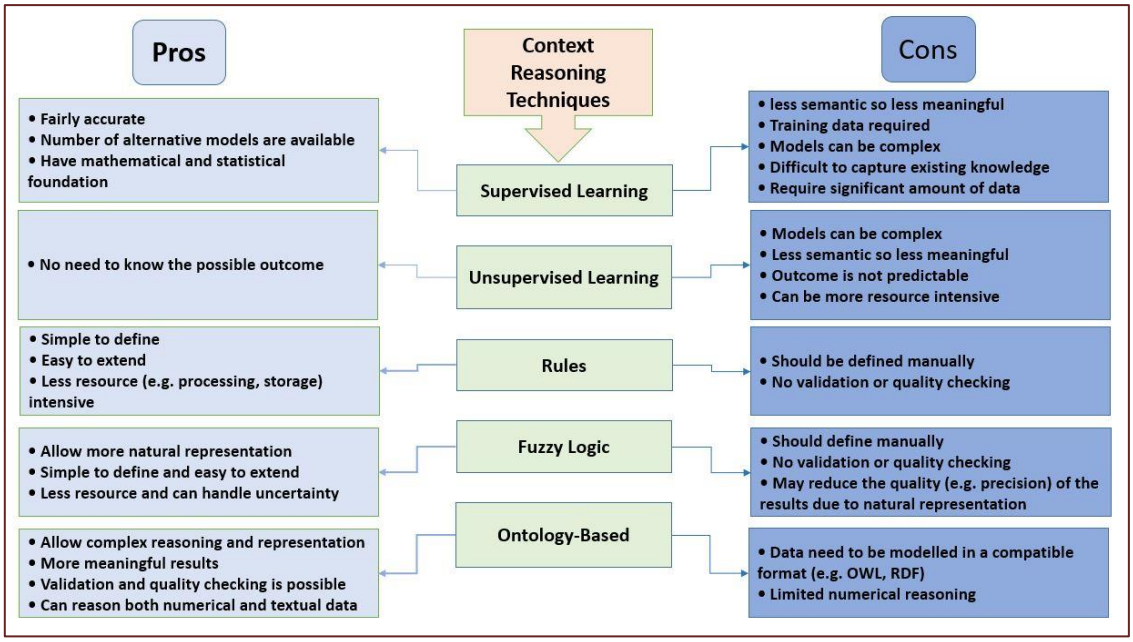


Figure 2-17: Comparison of context reasoning techniques (synthesised from Strang and Linnhoff-Popien, 2004; Perera et al., 2014; Cabrera et al., 2017)

In summary, several context-aware systems use context purely for decision-support/making or direct distribution services to the end-user. However, certain systems allow context information to be shared with other interested actors or

subsystems. This is defined as context information sharing, and represents one of the key challenges in the field of context-awareness for IoT (Perera et al., 2015; Boavida et al., 2016). Context information can be provided in various different ways, including variations in format, length, type and representation of the data (de Matos et al., 2020). Hence, there is a need to ensure that context-sharing platforms offer context interoperability. Thus, the following part of this chapter moves on to describe context information sharing through IoT platforms in greater detail.

2.4 Context Information Sharing through IoT Platforms and Middleware

Context-relevant data can be produced by IoT entities, and context management needs to be handled through a context management information processing layer. This layer would be expected to handle context data produced from multiple sources, including third-party software. Therefore, context-sharing functionality is facilitated by a context-sharing platform. Such platforms are capable of creating semantic interconnections between domains via the sharing of context information. As IoT environments can be highly complex, context-sharing platforms must be capable of dealing with a range of situations, and be able to implement service adaptation mechanisms driven by context-building blocks (de Matos et al., 2020). These building blocks can be categorised as properties and architectural components. The former applies to predominantly the software aspects of context-sharing platforms, including modelling, reasoning, dissemination, processing, interoperability, privacy, scalability and availability, as outlined in **Table 2-4**.

Table 2-4: Context sharing concerns

Context sharing Properties	Type	Aim	Implementation features	References
Modelling (M)	Properties	Responsible for mapping context into a predefined format.	Key-value, markup scheme, graphical, object oriented, logic-based, ontology-based and hybrid context modelling.	(Chen and Kotz, 2000; Perera et al., 2015)

Context sharing Properties	Type	Aim	Implementation features	References
Reasoning (R)	Properties	Defined as the process to obtain high-level information from less enriched, or even raw data	Supervised & unsupervised learning, rules, fuzzy logic, ontology-based, probabilistic reasoning.	(Bettini et al., 2010; Perera et al., 2015)
Data Dissemination (D)	Properties	The context information is shared to relevant entities	Static and dynamic.	(Perera et al., 2015)
Privacy (P)	Properties	Data on the context includes private data, such as User ID, preferences, activities, and location. Although these drive contexts, privacy preservation should apply.	Access control policies, anonymization, cryptography.	(Tiburski et al., 2015)
Interoperability (I)	Properties	Heterogeneity of data requires that different subsystems or systems must be interoperable	Interoperability through format, source, length, & representation, and semantic alignment	(de Matos et al., 2020)
Context Processing (CP)	Properties	It aims to obtain, produce, and share context information to service a data or service request	Searching, filtering, and aggregation.	(Lunardi et al., 2015)
Availability (AV)	Properties	The context must be always available for possible sharing	Availability ensured via cloud platforms or cached data	(de Matos et al., 2020)
Communication technologies (C)	Architectural Components	It refers to all equipment and programs that are used to process and communicate information	Communication devices, channels, and protocols for external and internal networks	(Doukas et al., 2015; de Matos et al., 2020)
History (Hi)	Architectural Components	Past data or inferred context stored locally or over the cloud.	Locally or cloud - based	(de Matos et al., 2020)
Architectural model	Architectural Components	Architecture can follow different patterns to support context sharing	Cloud-based, centralized-edge, and decentralized-edge	(de Matos et al., 2020)

Architectural considerations, regarding enabling hardware for the deployment of a context-sharing platform, include communication technologies, storage space and processing layers. In addition, some building blocks are strictly related to the context-sharing properties (e.g. modelling, reasoning and dissemination), including those that are specifically required in industrial monitoring for driving maintenance services adaptation. There are a variety of different IoT platforms, frameworks, services and middleware that are capable of collecting, processing and analysing sensor data. In this regard, various researchers (e.g. Perera et al., 2015; Mineraud et al., 2016; Sezer et al., 2018; Pradeep and Krishnamoorthy,

2019; de Matos et al., 2020) have examined surveys of such IoT platforms, frameworks, systems, prototypes, middleware and various different techniques. Some representative examples are listed in **Table 2-5**, which also shows their context modelling, reasoning and dissemination features.

Table 2-5: Comparison of context-awareness features of existing approaches

Approach Name	Year	Category	Modelling	Reasoning	Dissemination	Ref
Context Toolkit	2001	Toolkit	Key-value	(✓) Provided but not mentioned	Query	(Dey et al ., 2001)
Aura	2002	Middleware	Markup Schemes	Rules	Publish	(Garlan et al., 2002)
CoBrA	2004	Middleware	Ontology-Based	Rules, ontology-based	Query	(Chen et al., 2004)
MoCA	2006	Middleware	Markup Schemes, Ontology-Based	Ontology-Based	Publish, Query	(De Rocha and Endler, 2006)
DMS-CA	2008	System	Markup Schemes	Rules	Query	(Herbert et al., 2008)
CoSM	2009	Model	Ontology-Based	Ontology-Based	Dynamic	(Yamamoto et al., 2009)
FIWARE	2014	Platform	(✓) Provided but not mentioned	(✓) Provided but not mentioned	Query / Subscribe	(fiware.org/)
RCOS	2016	Middleware	Ontology-Based	Ontology-Based	Dynamic	(Dhallenne et al ., 2016)
PSW	2017	Model	Ontology-Based	Ontology-Based, Rules	Dynamic	(Ruta et al., 2017)
CoaaS	2018	Middleware	(✓) Provided but not mentioned	Rules, Pro	Dynamic	(Hassani et al., 2018)
SCENTS	2019	Middleware	(✓) Provided but not mentioned	Rules	Dynamic	(Liu et al., 2019)

Overall, this section has indicated that there has been significant interest in context-aware computing among scholars since its introduction around a decade ago. Although the sharing of context information between entities is a common objective of all of the presented projects, each has its own unique attributes, as they have various differences. All the presented projects are summarised in **Table 2-6**, which illustrates how they differ in the context-sharing building blocks,

and also indicates that the most frequently used techniques for the modelling feature are ontology-based. This is also applicable to IoT platforms, as ontologies enhance their support for interoperability.

A variety of different context-aware frameworks have been designed to show the advantages of this innovative technology, including the Context Toolkit, Aura and CoBrA, while additional frameworks remain under development. Everyday users are still unable to access context-aware services on a regular basis, however. Building context-aware systems remains a challenging process because of the absence of suitable infrastructure or middleware-level support. A suitable infrastructure for context-aware systems should have the ability to provide support for the majority of activities related to managing contexts, including: obtaining context from diverse sources, like physical sensors, databases and agents; interpreting context; disseminating context to relevant parties in a systematic and efficient manner; and generating programming models for the construction of context-aware services. In order to provide a supportive environment for the completion of these tasks, an effective context model should be established. Specifically, the majority of the above-mentioned platforms concentrate on the field of manufacturing, offering representations of manufacturing-field knowledge from a general perspective. Nevertheless, in terms of monitoring condition in manufacturing, they all lack a certain expressiveness with regard to representing knowledge about monitoring and maintenance activities.

A knowledge construct can be used to resolve context resolution requests in order to drive maintenance services. Such resolution can be achieved by ontological reasoning based on semantic similarity, determined through ontological distance metrics or other appropriate methods (Teoh and Case, 2004). This has relevance to similarity-based reasoning, such as that typically used in CBR systems, which have been employed in the maintenance domain in the past (Cândea et al., 2014). However, modelling and reasoning capabilities in ontologies go beyond CBR similarity. For OWL2-based reasoning, the formulation of queries can be done via SPARQL queries in RDF documents.

Additionally, depending on the complexity of a given ontology model, the process of semantic matching can be served using SWRL. Overall, there is a need to further develop ontology-based modelling and inference in order to drive maintenance services by extending currently-employed ontology concepts to include the key additional and operational ones that are typically included in the relevant standards, but less so in the literature. In this regard, the next section moves on to describe in greater detail of ontologies in predictive maintenance and asset management. This is important in order to develop a maintenance context ontology for the framework focusing on modelling failure analysis of mechanical components.

2.5 Ontologies in Predictive Maintenance and Asset Management

As the manufacturing environment becomes more knowledge-intensive and dynamic, maintenance is becoming more and more crucial in asset lifecycle management. The use of semantic technologies, particularly ontology-based modelling for predictive maintenance, has become an important research topic, and thus many ontologies have been offered to promote knowledge representation and reuse within the context of predictive maintenance. In this regard, various different ontological modelling approaches have been pursued in the fields of production, maintenance and asset management over the years. Using ontologies to model domain knowledge is a valid approach, and therefore several research efforts utilising or recommending ontologies in the domain of asset and maintenance management have been reported in the literature. Some of these sought to adopt relevant standards as a baseline for the ontology concepts; for example, an asset management ontology based on the PAS55 recommendation, which was later subsumed by the ISO 5000 standard, has been reported (Frolov et al., 2010), but the scope is broader than simply maintenance and, while it serves asset management purposes well, it does not specifically target maintenance.

When considering maintenance in the manufacturing domain, it is of interest to capture the functional impact of the asset integrity level on the actual production

process. The anticipated integrity level would then require a predictive approach (Cao et al., 2019). Although such an approach can be highly relevant, prediction based only on historical data, without accounting for predicted future operating aspects, is appropriate only insofar as the historical data align with future expectations, which is often not the case. However, if the intended use of an ontology is to serve maintenance action determination, planning and scheduling, then operational semantics need to be included in the modelling.

Appropriate knowledge constructs with potential impact, which link assets, their function and their faults, are failure modes and effects analysis (FMEA) or failure mode, effects and criticality analysis (FMECA) (Nuñez and Borsato, 2018). An FMECA approach would still be limited, however, in that, while it associates assets and component faults with detectability, it does not include explicit information about measurement methods per asset and fault type, or specific measured parameters for the measurement methods. While this is appropriate for the original intended purposes of a FMECA study, it falls short of the requirements for a knowledge formalism that would serve operational purposes.

A more promising approach would be to extend the FMECA by including link-failure modes in the ontology concepts, with more detailed diagnostic information and associated recommended actions to the resources that would be needed for implementing the actions, such as spare parts and human resources (Jin et al., 2009). Such an extension could look into the recommendations of relevant standards (ISO 13374:1, 2003; ISO 17359, 2011) that link monitoring parameters and fault indicators to failure modes and recommended actions (D'Elia et al., 2010). A maintenance ontology may comprise multiple layers—an upper-level ontology to abstract the key domain concepts and a lower-level ontology contextualised around specific operational factors (Koukias et al., 2013).

Monnin et al. (2011) developed a knowledge model for fleet predictive maintenance to handle fleet-wide contextual knowledge, arguing that fleet-wide Prognostics and Health Management (PHM) requires a knowledge-based system capable of handling contextual information. Decision-making processes for diagnosis and maintenance are strengthened by semantic modelling, which deals

with the definition of concepts and the relationships between them. As another example, an ontology was developed for predictive maintenance in the wind energy domain, being used as a basis for the identification and diagnosis of faults in monitoring the condition of wind turbines (Papadopoulos and Cipcigan, 2010). The proposed ontology model was used by conducting ontology queries to detect potential failures and their specific locations in the gearbox of the wind energy converter.

When considering the manufacturing domain, it is most useful to capture the level of functional impact of asset integrity on the actual manufacturing process. Castet et al. (2018) presented an approach for capturing fault information in a modelling environment using the ontology of fault management and a set of plugins designed to automatically extract two reliability artefacts—the FMECA and fault tree. The FMECA offers a sound basis upon which to express the organisational and functional association between a manufacturing asset hierarchy and its linkage with the functional integrity of the production facility. Nuñez and Borsato (2018) proposed an ontological model called OntoProg that served as a widely accepted data and knowledge representation scheme for diagnostic-oriented maintenance, capable of being used in different types of industrial machines. They also suggested a set of SWRL rules to improve the ontology's expressiveness. More recently, Cao et al. (2019) introduced an ontology-based approach to facilitate predictive maintenance in industry. This approach combines the use of blurry clustering and semantic technologies. Ontological approaches that employ industrial scenarios to support maintenance management have been developed for a range of assets, including urban infrastructure (Wei et al., 2020), highway infrastructure (France-Mensah and O'Brien, 2019), building information management (Farghaly et al., 2019), transport infrastructure (Ren et al., 2019; Li et al., 2020) and railway infrastructure (Dimitrova et al., 2020). **Table 2-6** summarises other studies on ontologies in maintenance and asset management.

Table 2-6: Ontologies used in maintenance and asset management

Survey Title	Ontology domain	Contribution	Reference
"A Formal Ontology for Semantics in Maintenance Platforms"	Production system	An ontology to produce new knowledge in the field of industrial maintenance.	(Karray et al., 2012)
"Ontology-Based Implementation of an Advanced Method for Time Treatment in Asset Lifecycle Management"	Lathe Machine	Implemented a method for exploiting the characteristics of time in maintenance asset lifecycle management (ALM) systems.	(Matsokis and Kiritsis, 2012)
"Ontology-Based Schema to Support Maintenance Knowledge Representation With a Case Study of a Pneumatic Valve"	Pneumatic Valve	A methodology for knowledge representation using ontological principles is proposed.	(Ebrahimipour and Yacout, 2015)
"A novel maintenance system for equipment serviceability improvement "	Manufacturing machine	A maintenance system for real-time equipment that integrates augmented reality (AR) for context-aware overlay of textual and graphical maintenance instructions on the maintenance scene.	(Ong and Zhu, 2013)
"Context-Aware Recommendation for Industrial Alarm System"	A power generation plant	Via semantic web technology and machine learning techniques, an industrial alarm management system is suggested.	(da Silva et al., 2018)
"Semantic Data Model for Operation and Maintenance of the Engineering Asset"	N/A	A semantic data model for engineering asset management is proposed.	(Koukias et al., 2013)
"Context Modelling with Situation Rules for Industrial Maintenance"	knowledge gateway system	A knowledge modelling approach is proposed to support maintenance personnel	(Aarnio et al., 2016)
"A research on intelligent fault diagnosis of wind turbines based on ontology and FMECA"	Wind turbine	A method based on ontology and FMECA for intelligent wind turbine fault diagnosis is suggested.	(Zhou et al., 2015)
"Building an ontological knowledgebase for bridge maintenance"	Transport infrastructure	An ontology for achieving automatic control of rules and improving the management and communication of bridge maintenance knowledge.	(Ren et al., 2019)

A review of the related research work revealed two issues. Firstly, there is a missing link between knowledge constructs and operational, reliability-based services adaptation actions. Focusing on the asset context, relevant domain knowledge can be modelled in many forms, but of particular interest are knowledge constructs relevant to reliability analysis, such as failure modes, effects (and criticality) analysis (FME(C)A), FMEA or FMECA models. These are, however, often utilised in design-stage engineering studies. Maintenance

services, on the other hand, need to be invoked during the operating time, and so relevant information representations need to be enriched in order to enable dynamic context inference and the composition of contextually-relevant services.

Secondly, existing predictive maintenance approaches in the manufacturing domain are still limited to the deployment of condition monitoring systems for identifying the failure mode and effects analysis in mechanical components. Therefore, the resolution of asset context is needed to analyse mechanical systems, and to logically connect measurements, observed behaviour and intended function, with machinery operating condition and faults. To this end, FMEA offers an appropriate grounding for the baseline of the knowledge mapping. According to ISO 17359 (2011), failure mode analysis based on FME(C)A is recommended so as to ensure that maintenance activities are consistent with established fundamental practice-oriented knowledge.

2.6 Chapter Summary and Research Gaps

The literature review offered important insights into the lifecycle of contexts, which determines where data are generated and where they are consumed. During this lifecycle, context awareness can be regarded as a service, defined by various researchers as ‘context as a service’.

While all the approaches deal with some form of context management, starting with acquisition and modelling, eventually the actionable context needs to be domain specific. In the application domain of asset and maintenance management, context strongly depends on the assets and their hierarchy. Unless such context is captured, it is difficult to convert IoT-generated data from industrial systems to actions. Therefore, it is important to create a representation that integrates qualitative and quantitative data, wherein data and service delivery is determined upon resolving the apparent context of a service or data request. The most common approach to achieving this is through ontology-based modelling. This allows a more meaningful representation of context and supports semantic reasoning. In addition, it can overcome several technical restrictions, such as interoperability and strong validation and expressivity.

The advantages and disadvantages of each context lifecycle management method are distinctly observable. Ontology-based reasoning appears to be effective at deriving high-level context from existing contexts for remote monitoring services because of the dominance of knowledge sharing, logical inference and the reuse of knowledge. Furthermore, it allows the context information to be stored according to the ontological structure, and automatically reasons later, when required. In remote monitoring services, this can enable not only fault detection, diagnostics and prognostics, but also action recommendations consistent with the inferred context of the analysed situation. In addition, rules represent the most uncomplicated and easiest techniques for reasoning, out of all those available. The structure of rules generally follows the if-then pattern, permitting the creation of upper-level content data using lower-level context (Kessler et al., 2009; Zhou et al., 2009).

2.6.1 Research Gaps

The following research gaps were identified from the literature review.

Table 2-7: Link gaps with research questions and sections

Research gap section	Research gap	Research question
2.5	Asset and maintenance management are concerned with the management practices, technologies and tools necessary to maximise the value delivered by physical engineering assets. IoT-generated data are increasingly considered to be an asset, and their data asset value needs to be maximised, too. However, asset-generated data, in practice, are often collected in non-actionable form, and need to be modelled and represented to ensure that they continue to function properly, and that their value is optimised over their lifecycle (Perttunen et al.,2009; Bettini et al., 2010; Perera et al., 2014; Collins and Lanz., 2019; Al-Shdifat et al., 2020).	How can context be acquired, modelled, processed and disseminated for industrial monitoring services?
2.2.5	The IoT has expanded the range of applications with substantial needs for context management, and this is reflected in the focus of the relevant studies. Nonetheless, while substantial research efforts have been devoted to context lifecycle management in web-based, mobile and ubiquitous computing, including IoT-enabled computing, little attention has been given to translating these advances into tangible progress in industrial monitoring services (Lee and Martinez Lastra, 2013; Scholze et al., 2017; Al-shdifat and Emmanouilidis, 2018).	What is an appropriate framework to manage context awareness in a way that facilitates efficient condition monitoring?
2.3.4.1 & 2.5	The most applicable context-modelling techniques that have been examined are ontology-based. However, the studied approaches lack some expressiveness concerning the representation of knowledge for monitoring services in manufacturing environments (Castet et al. 2018; Nuñez and Borsato 2018; Al-Shdifat et al., 2020).	How can context be acquired, modelled, processed and disseminated for industrial monitoring services?
2.4 & 2.2.5	There is a need to develop a framework and an architecture that can introduce context awareness to enhance monitoring services in manufacturing environments	What is an appropriate framework to manage context awareness in a way that facilitates efficient condition monitoring?

This chapter included a thorough, detailed review of context awareness in the IoT. The following chapter presents a discussion on the research methodologies that were considered for this study, offering a justification for the ones ultimately applied to answer the RQs in order to achieve the ultimate research aim.

3 RESEARCH DESIGN AND METHODOLOGY

3.1 Introduction

This chapter outlines the methodological framework developed to address the ROs. In presenting this methodological framework, the philosophical and theoretical perspectives underpinning the development of the context-aware IoT framework for industrial monitoring services are examined, and the multi-phase methodological approach adopted is discussed. This study was aimed at developing a flexible and more effective framework in a way that would facilitate efficient industrial monitoring. To achieve this, the intention was to solve the problem theoretically by developing a framework, architecture and maintenance ontology, and validating these through laboratory experiments as part of a case study and expert judgments.

First, the design of the study is presented and its philosophical underpinnings explained. The design is typically the means by which researchers address their primary RQs. It is determined by the relative value the researcher ascribes to different research methods (Bryman and Bell, 2011). Design therefore dictates which data collection and analysis methods are applied, and the validity of the subsequent findings. "Research methods include all the techniques and methods which have been taken for conducting research where as research methodology is the approach in which research troubles are solved thoroughly. It is a science of studying how research is conducted systematically" (Mishra and Alok, 2011). Thus, the selection of a methodology must be underpinned by the adoption of a wider research paradigm. The next section explains and describes a variety of philosophical underpinnings to research. This is followed by a rationale and justification for the philosophical stance assumed for this study and the research methodology adopted.

3.2 Research Philosophical Approach

Saunders et al. (2009) likened the process of research to an onion, with the research steps being the individual layers. Assumptions are needed at each step regarding the design of the research and its associated methodology. There are

six key elements in research—philosophies, approaches, strategies, choices, time horizons, and techniques and procedures employed in the data collection and analysis. Subsequent subsections define the key philosophical assumptions the researcher must address. This is followed by a brief discussion of the six core research elements.

3.2.1 Research Philosophical Assumption

In essence, a research philosophy comprises the core beliefs and assumptions held by the researcher at each step of the research process (Burrell and Morgan, 1979). To develop a theory, the researcher needs to be cognisant of the philosophical basis of their research. It is therefore essential to clarify the ways in which the core philosophical assumptions–ontology, epistemology and axiology differ. These assumptions are depicted in **Figure 3-1**.

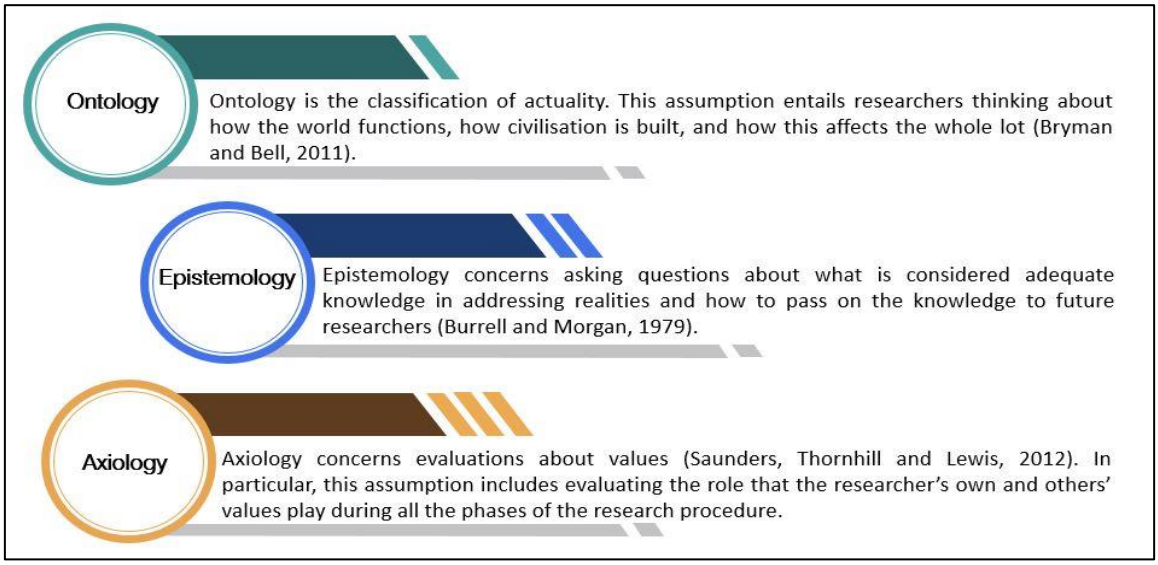


Figure 3-1: Research philosophical assumptions

3.2.2 Philosophical Stance of this Research

There are five philosophical stances in the first layer of the research onion–positivism, critical realism, interpretivism, postmodernism and pragmatism. These are summarised in **Appendix A**. To recall, the main objective of this study was to analyse the current practices of context lifecycle management, and identify the appropriate factors that are used in context acquisition, modelling, reasoning

and dissemination for IoT-enabled industrial monitoring services. The outcome of this objective is to be used in developing a framework that introduces context awareness to enhance remote monitoring services in manufacturing environments, and in developing a maintenance ontology for the framework, focused on modelling the failure analysis of mechanical components. To achieve this, the researcher intended to solve the problem theoretically, by developing a framework and validating it through laboratory experiments and a case study. Given this, pragmatism was adopted as the philosophical stance in this study. The aim of pragmatism is to provide useful solutions to the problem that the research is addressing (**Appendix A**). This practical solution will aim to shape the way procedures are undertaken in the future in order to produce the best results.

3.3 Research Approach

Having selected a philosophical stance, it was important to then decide upon an appropriate approach to the research. Three disparate approaches–deductive, inductive and abductive–were considered. When the aim is to establish a causal relationship between variables, a deductive approach is typically employed. This was defined by Saunders et al. (2018) as an approach based on theory-testing, where the aim is to verify the validity of a certain set of assumptions. Conversely, an inductive approach involves generating and building new theories (often in the form of a conceptual framework) (Bryman and Bell, 2015). The third approach–abductive research–concentrates on the empirical explanation of research questions. Based on the stated aim and objectives, the approach adopted in this study was an inductive one. Since this research sought to identify the appropriate factors that are used in context lifecycle management for IoT-enabled industrial monitoring services, and based on the stated aim and objectives, the inductive approach was adopted.

The next step in the research process is to verify its purpose. In this respect, there are four possible alternatives: explanatory, exploratory, correlational, and descriptive, as presented in **Figure 3-2**.

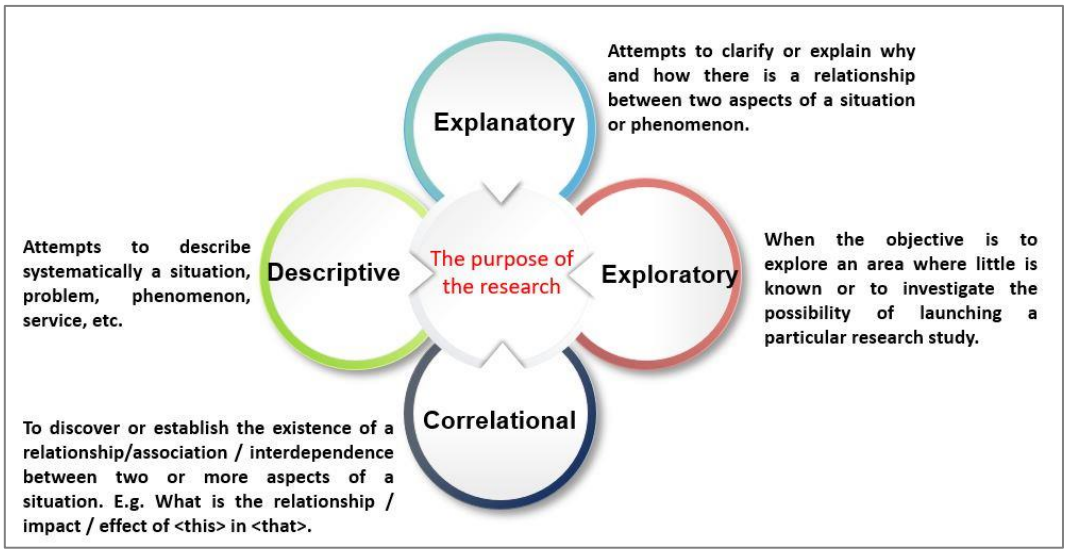


Figure 3-2: Types of research – Viewpoint of objectives (adapted from Kumar, 2005)

For this study, an exploratory approach was appropriate to satisfy the aim of investigating the unexplored gap between the integration of IoT, cloud computing and context awareness for industrial monitoring services, as identified in Chapter 2.

3.4 Research Choice

A key element in every research study is the process of data collection. Such data can be either quantitative, and therefore collected through questionnaires, surveys or as secondary data, or qualitative, and therefore collected through interviews and documentation (Creswell, 2009). Equally important are the techniques employed to analyse the data. If the data is quantitative, statistical analyses are usually conducted. If they are qualitative, a range of techniques can be applied, such as thematic analysis or grounded theory. Making the correct methodological choices with respect to data collection and data analysis is therefore imperative. In terms of whether to adopt a quantitative or qualitative approach, there are three possible options—the monomethod (quantitative or qualitative), multimethod (quantitative and qualitative) and mixed method. This study fell into the mixed-method category. This choice meant that the research would include both quantitative and qualitative procedures and techniques to collect and analyse data. To develop the theoretical framework, secondary data

were collected, comprising peer-reviewed articles in high-quality journals with a high impact factor, books, reports and conference proceedings. To conduct the case study, data were collected from the operational and maintenance records of a laboratory-based test rig. In addition to case study validation, experts' judgment on the developed maintenance context ontology for the framework was sought. In this respect, a basic questionnaire was developed to be administered to the participants to determine the extent of their knowledge regarding ontologies, faults, causes and effects of failure along with maintenance terminology.

3.5 Research Strategy

The research strategy is an overall plan of action adhered to by the researcher to address the RQs and fulfil the ROs (Saunders et al., 2012). The exact strategy employed will depend on the ROs, the findings of previous research, the time available and, above all, the resources and data available. The seven principal strategies are "experimentation, survey, case study, action research, grounded theory study, ethnography and archival research". For this study, the intention was to develop a framework, and validate it through laboratory experiments and case studies. Therefore, given the objectives of this research (as stated above), an experimental and case-study strategy was deemed the most appropriate to adopt. This is explained in further detail in Section 3.7.

3.6 Time Horizon

The fifth layer of the research onion is the time horizon. This is a vital component of research, and is associated directly with the RQs (Saunders et al., 2012). It concerns whether the research is carried out at a specific time, in which case it will be cross-sectional, or over a longer period, in which case it will be longitudinal. This study required a longitudinal time horizon, the aim being to investigate changes, and the developments that take place, in specific phenomena over time. Although time constraints can create problems for this type of study, these is offset by the potential value of the insights obtained. However, if secondary data is required, the availability of such data may become another constraining factor (Saunders et al., 2012).

3.7 Research Structure Diagram

The methodological structure of this study was shaped by both the aim and objectives of the research, and the questions it sought to answer. These required the adoption of appropriate research tools and methods (Creswell and Poth, 2016). The methodological contribution made by this research is that it was intended to highlight the multi-phased strategy needed to develop robust research frameworks with high validity. This framework, depicted in **Figure 3-3**, comprised four phases, all of which made a distinct contribution to the study. Initially, this study's function was to recognise its aim as well as its objectives. A study of the RQs was undertaken, in addition to the research area. Furthermore, additional enhancement of the objectives may have been achievable by means of a literature review, with technical development requirements in the research area as well as the research direction's new outcomes. **Figure 3-3** illustrates all the phases that helped in proceeding with an initial estimation of the overall timing and milestones. The information related to each phase is briefly explained below.

3.7.1 Phase 1 – Establish the Research Baseline

The initial phase of this study paid particular attention to the contextual understanding and comprehension of the ongoing practices of context lifecycle management and the integration of IoT and cloud computing for industrial monitoring services. An extensive literature review was conducted to examine related work and determine factors, methods and techniques appropriate for the study, in order to understand the latest developments and to identify gaps in the research area (**Objective 1**). In light of that, the appropriate factors used in context lifecycle management were critically analysed and published in a conference paper.

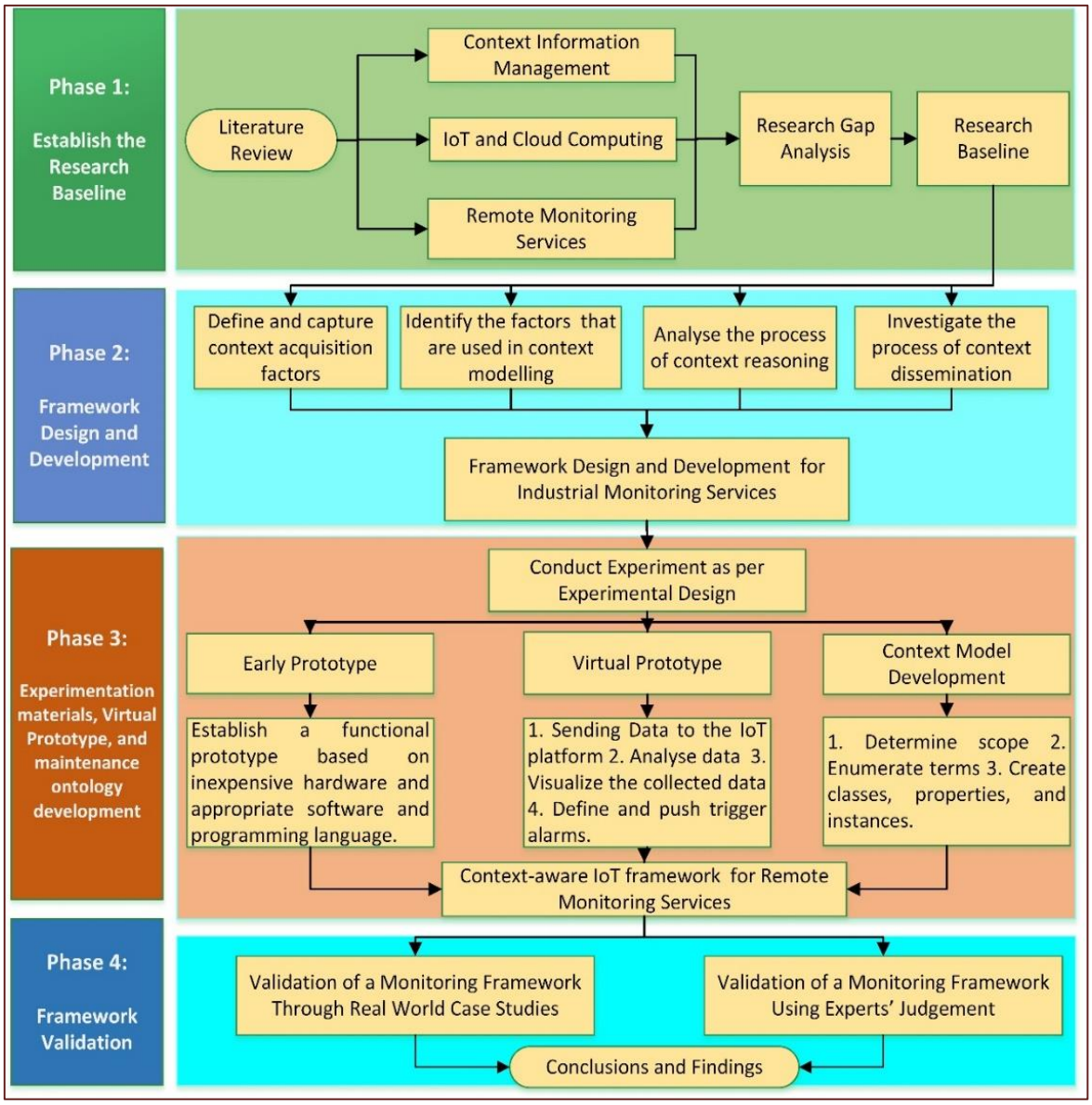


Figure 3-3: Research methodology

3.7.2 Phase 2 –Framework Design and Development

This phase involved the design and development of a framework and an architecture that would introduce context awareness to enhance remote monitoring services in manufacturing environments through a set of steps. Firstly, the context acquisition factors were defined and captured. Secondly, a representation mechanism appropriate for context modelling was identified. Finally, the process of context reasoning and dissemination, in order to derive high-level context deductions from a set of contexts, and to deliver context to end-

user applications, was investigated (**Objective 2**). This is explained in further detail in Chapter 4. In this regard, the initial framework and architecture were developed after reviewing relevant and commercial solutions. The initial framework has been published in a conference paper.

3.7.3 Phase 3 – Experimentation Materials and Methods

The third phase involved the early prototype, virtual prototype and context model development (**Objectives 2 and 3**). For the early prototype, a functional test was performed in order to investigate the functionality of the key components of the proposed framework and architecture. In this regard, physical implementation was conducted from data generation to data visualisation because the problems, benefits and challenges needed to be understood before a real-world case study was undertaken, as outlined in **Figure 3-4**.

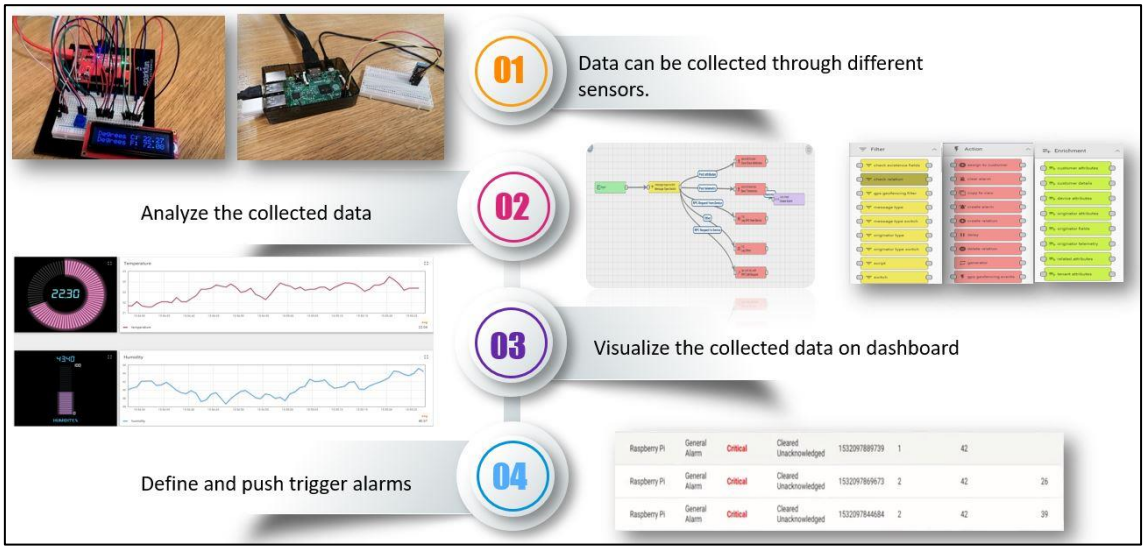


Figure 3-4: Early Prototype phase

Data are collected through different sensors and are fed into an embedded computing device. The device is networked and is able to push the data to a cloud platform where they can be processed and visualized into relevant information for the users. That information can be presented in different forms; the most commons are simple monitoring of the raw data on graphs to generate suggestions for possible interventions based on the measurement ranges. When

unexpected or unwanted events are detected, the system issues alerts, which are communicated to users or other systems.

For the virtual prototype, a gearbox test rig was chosen as a case study to validate the applicability of the proposed framework and architecture. In light of that, different scenarios have been created. These scenarios focused on the design of simulation through two different scenarios ("Building Management System" and “Machine Monitoring System”) to illustrate the operation and utility of the proposed architecture for the integration of IoT and cloud computing for industrial monitoring services as shown in **Figure 3-5**. In order to do that, ThingsBoard platform was utilized to validate the applicability of the proposed framework and architecture. ThingsBoard is an open-source IoT platform that allows IoT projects to be rapidly developed, managed and scaled. The platform enables users to develop rich IoT dashboards for the purpose of visualising data and then applying controls on remote devices in real-time. ThingsBoard facilitates the connectivity between devices through developed application layer protocols ThingsBoard (e.g. MQTT, CoAP and HTTP) and provides for deployments via the cloud.

Figure 3-5: Virtual Prototype phase

For the context model development, the focus of the maintenance ontology is on modelling failure analysis of mechanical components to answer queries regarding how faults manifest themselves and how they can be prevented or addressed, so

as to adapt relevant diagnostics or maintenance actions in a Condition-Based Maintenance setting. This is explained in further detail in Chapter 5.

The development of ontology can be based on one of the numerous procedures described in the literature (Sure et al., 2009), including Uschold and King, Grüninger and Fox, Methodology, Ontology Development 101 (OD1) and KACTUS. In the current research, the Ontology Development 101 is adopted due to the following reasons: (1) This methodology was designed for beginners. As such, it is easy to learn and operate. (2) The detailed activities involved in this approach have been specified. The process of establishing an ontology is described in detail in this methodology. (3) It can be integrated with other tools. This method contains detailed instructions on how to implement the ontology in the Protégé environment (Ren et al., 2019). OD1 comprises six stages (Noy and McGuinness, 2001). However, along with identification of the procedure that will be adopted, the development of ontology models requires tools that can support all activities in the development process. Such tools include TopBraid (www.topquadrant.com) and OntoStudio (www.semafora-systems.com), as well as open ones, such as the popular OntoEdit (Sure et al., 2002), HOZO (Kozaki et al., 2005) and Protégé (https://protege.stanford.edu/). Specifically, Protégé is the most dominant application (ontology publisher) due to the fact that it is an open platform that offers Plug-in extensibility as well as XML (S), OWL, RDF (S) and Excel along with graphic taxonomy, queries in SPARQL, rules in SWRL language, and a reasoner (Pellet).

3.7.4 Phase 4 – Framework Validation

In the final phase of the research, the results had to be validated to test their applicability, and to identify areas for further development. The selected gearbox test rig case was employed to validate the applicability of the proposed framework and ontology. In this regard, several ontology evaluations have been proposed that could take an implementation or a design viewpoint (Degbelo, 2017; Kumar and Baliyan, 2018). In the validation process, the semantic and syntactical correctness of the ontology is ensured and verified, whether the ontology satisfies the targeted requirements or not. The scope of the present case study was

exploratory; that is, the aim was to present the development of a context-resolution service mechanism for industrial diagnostics, based on the design of a maintenance ontology focused on modelling the failure analysis of mechanical components. Therefore, it was considered appropriate to focus on a subset of evaluation criteria–namely robustness, level of detail, effectiveness, internal consistency and applicability–within the viewpoint of the targeted application case study. In addition to case study validation, experts' judgment on the developed maintenance context ontology for the framework was sought. This is explained in further detail in Chapter 6.

3.8 Chapter Summary

In this chapter, the research method used in this study was explained, with **Figure 3-6** summarising the methodology selection.

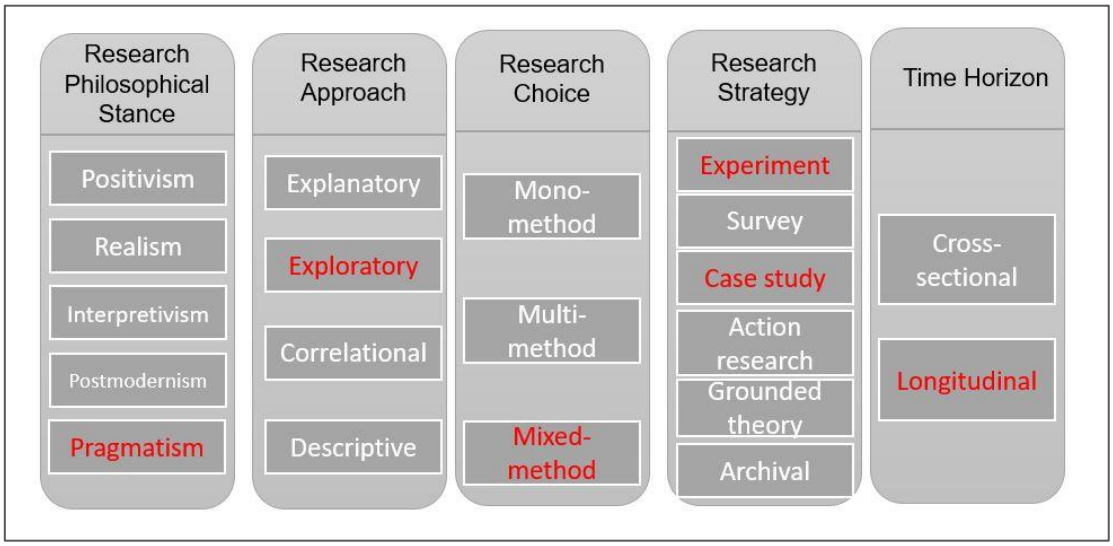


Figure 3-6: The research methodology selection (adapted from Saunders et al., 2018)

In this chapter, the common research philosophical approaches followed were defined, including the philosophical research assumptions, philosophical stances, approaches, strategies, choices and time horizons. In each element of the philosophical research approach, different options were evaluated, and the position of the present study was highlighted among the available options. The aim of the study was to design and implement a context-awareness framework

and an architecture for the integration of IoT and cloud computing for industrial monitoring services. Consequently, pragmatism was adopted as the philosophical stance. Furthermore, based on the stated aim and objectives, the approach adopted was an inductive one. The research fell into the mixed-method choice, meaning that the study included both quantitative and qualitative procedures and techniques in order to collect and analyse the data. Finally, given the objectives of this study (as stated above), an experimental and case-study strategy was deemed the most appropriate to adopt, and the overall structure of the research was illustrated.

Research ethics were considered throughout the research process. In accordance with the General Data Protection Regulation, all data has been handled in a way that preserves confidentiality and privacy. The effective employment of this methodological framework enabled the attainment of the research aim–the development of a context-awareness framework for industrial monitoring services. The insight gained from the second phase of the multi-phase research methodology–architecture and framework development–is discussed in the next chapter.

4 ARCHITECTURE AND FRAMEWORK DEVELOPMENT

4.1 Introduction

The introduction of IoT technologies has expanded the ability of industries to generate data from devices that can sense and communicate in real time, supporting decision-making processes for monitoring the state of equipment, and offering guidance for proactive maintenance (Bousdekis et al., 2015). The explosive growth of IoT-generated and -managed data requires substantial further effort for the effective management and exploitation of the data, however. The absence of general architectural knowledge is currently preventing researchers from exploiting the scope of IoT-centric approaches. Due to distinct methods, standards and use cases, insufficient focus has been applied to translating these developments into actual progress in terms of industrial monitoring services. Resultantly, there is a need to develop architectures and frameworks that can introduce context awareness in order to enhance monitoring services in manufacturing environments.

This chapter comprises two core sections. Section 4.2 presents an overview of architecture development for the introduction of context awareness to enhance remote monitoring services in manufacturing environments. This consists of four viewpoints–business, usage, functional and implementation. Section 4.3 describes the development of a context-aware IoT framework for remote monitoring services. The framework applies context-aware computing to deliver solutions and address key challenges that IoT-enabled monitoring services need to handle, specifically how the context can be modelled, processed and disseminated for remote monitoring services, how this impacts the service discovery solution, and what an appropriate taxonomy of ontology is. This represents phase two of the multi-phase research methodology presented in Section 3.7 and RO 2.

4.2 Architecture Development

Viewpoints represent the central aspects of the ISO/IEC/IEEE architecture description standard (ISO/IEC/IEEE 42010:2011). A viewpoint consists of conventions that frame the analysis and description of particular system concerns. One or multiple concerns are framed by a viewpoint. A concern is defined as any topic of interest associated with the system. A stakeholder refers to a person, group, organisation or categories thereof who have a stake in a particular concern, and hence have an interest in the given viewpoint and system. In line with the approach stipulated in the ISO/IEC/IEEE architecture description standard, the concepts used to describe, analyse and resolve the group of particular concerns in each viewpoint are defined as that viewpoint's architecture. In this regard, the proposed architecture, to introduce context-awareness to enhance remote monitoring services, has been developed based on the Industrial Internet Reference Architecture (IIRA) and is actively maintained by the Industrial Internet Consortium (IIC, 2017). This architecture comprises four viewpoints, as illustrated in **Figure 4-1**. These viewpoints are business, usage, functional and implementation.

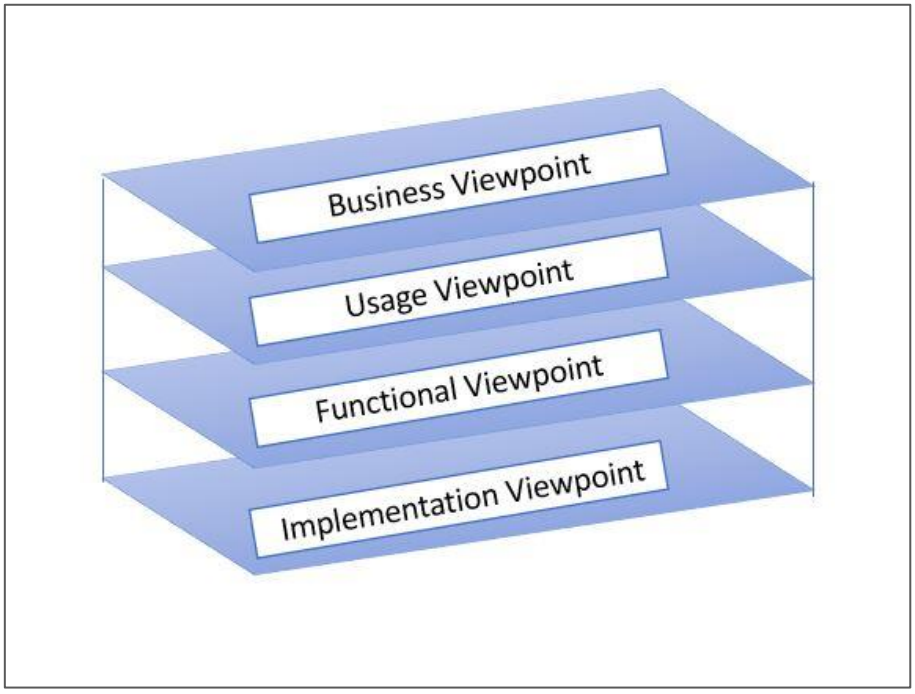


Figure 4-1: Architecture viewpoints (adapted from IIRA 2015; IIC, 2017)

4.2.1 The Business Viewpoint

The business viewpoint is concerned with identifying users, as well as their business vision, values and goals (IIC, 2017). Such concerns are particularly interesting to product managers, systems engineers, technicians and decision-makers. Business-focused concerns, such as business value, maintenance costs, increased efficiency, machine availability and product liability, must be assessed in regard to the application of IIoT systems for resolving business issues.

Focusing on the application domain of asset and maintenance management, context information management was considered on the basis of a valid knowledge construct for reliability-oriented maintenance management, wherein data and service delivery were determined through resolving the context of a service or data request. Therefore, the main business viewpoint goal was to increase efficiency and availability, make informed decisions, minimise costs and maximise profits, all without compromising the service or product quality. When users interact with systems in this regard, the proposed maintenance ontology can help them to narrow down the list of options to those that are contextually relevant. This allows both managers and operators to assume control and make more effective decisions so that general efficiency can be enhanced throughout the entire industry. Furthermore, the utilisation of machine monitoring systems enables machines to monitor problems that occur in their operations, such as the failure of components or outages, and then report them to the manager so the required fixes can be applied. **Figure 4-2** presents a proposed high-level view of the viewpoints of key stakeholders, and the key concepts through which these can be served.

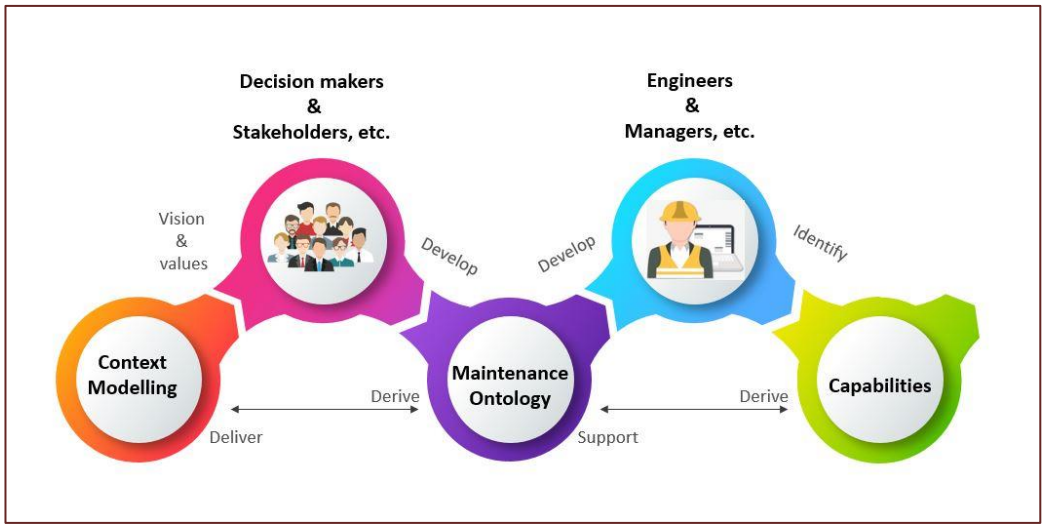


Figure 4-2: Business viewpoint

- Stakeholders have a significant interest in the business and are highly influential in its direction. They are frequently considered to have strategic thinking skills and are innovators in a firm or an industry sector. Employees from various departments are involved in the maintenance and asset management function, including those with engineering, technical, managerial and financial duties. Data and knowledge applicable to certain staff roles may be irrelevant to others. Maintenance and asset management information systems must account for such variations in the roles and responsibilities of different staff. The effective adaptation of services and data provisioning is sought via context-adaptive computing, which is becoming more relevant to enterprise information systems.
- Vision determines the direction in which the business applies its focus. Upper-level business stakeholders generally formulate and distribute the vision of the organisation. In this study, context management was considered on the basis of a valid knowledge construct for reliability-oriented maintenance management. The vision was to produce a semantic organisation of data so as to drive maintenance services.
- Values mirror the manner in which the vision can be viewed by the stakeholders who provide the funds for the application of the new systems, and also by the users of the implemented system.

- Key objectives are quantifiable and high-level technical, and are the ultimate business results required of the implemented system in terms of the delivery of values. In this study, the collected data involved a wide number of parameters, gathered over long periods of time, and were of significant scale. However, they may potentially have failed to represent the range of possible scenarios of asset operation, or the causal relationships between the monitored parameters, and so the size of the data collection, while adding to the complexity of the problem, would not necessarily allow direct data-asset-value exploitation. One way to handle data complexity is to introduce context information modelling and management, wherein data and service delivery are determined upon resolving the apparent context of a service or data request. Moreover, a reasoning mechanism can be used for delivering context resolution, where a metadata layer can be added by the determined context resolution onto data or events produced via automated and manual methods.
- Fundamental capabilities are defined as the high-level specifications of the system's basic ability to conduct specific important business activities. Primary goals represent the foundation for identifying fundamental capabilities. In this research, remote monitoring solutions facilitate the process by which the performance and functionality of remote equipment and installations are monitored, controlled and supervised by users. Remote monitoring gives users the ability to continue viewing and controlling equipment, facilities and operations in locations that are unsupervised, difficult to access and in isolated locations.

4.2.2 The Usage Viewpoint

The usage viewpoint is related to the manner in which an IoT system achieves the primary capabilities determined in the business viewpoint (IIC, 2017). This viewpoint delineates the activities that provide coordination for different work units over different system elements. Such activities, which describe how the system is utilised, act as an input for the system requirements incorporating those denoting important system properties and also facilitate the processes of

designing, implementing, deploying, operating and developing the IoT system. Moreover, the usage viewpoint demonstrates how applications function in harmony for the purpose of supporting processes within the business. It can be employed for identifying the services required by such processes and different applications, or in developing business processes through describing the services available.

In the application domain of asset and maintenance management, context is relevant to the asset and its hierarchy, the user, the production or service business circumstances, as well as the overall system and operating-environment aspects (Emmanouilidis et al., 2019). The resolution of asset context is needed to analyse mechanical systems, and logically connect measurements, observed behaviours and intended functions with machinery operating condition and faults. To this end, the usage viewpoint relates to the application of engineering concepts in the context of optimising equipment, procedures and departmental budgets to better maintain equipment and its reliability and availability. **Figure 4-3** illustrates a workflow of the usage viewpoint. Initially, information pertaining to the maintenance, evaluation and decision support of machines is translated by knowledge-engineers (analysts) into ontology and SWRL rules. Next, Protégé and SWRLTab are used to define the ontology and SWRL rules, which are then stored in a knowledge base. Subsequently, SWRL rules are executed by the rules engine, which produces new facts within the ontology management system. Lastly, decision-makers can acquire beneficial information by applying multiple constraints via the query interface.

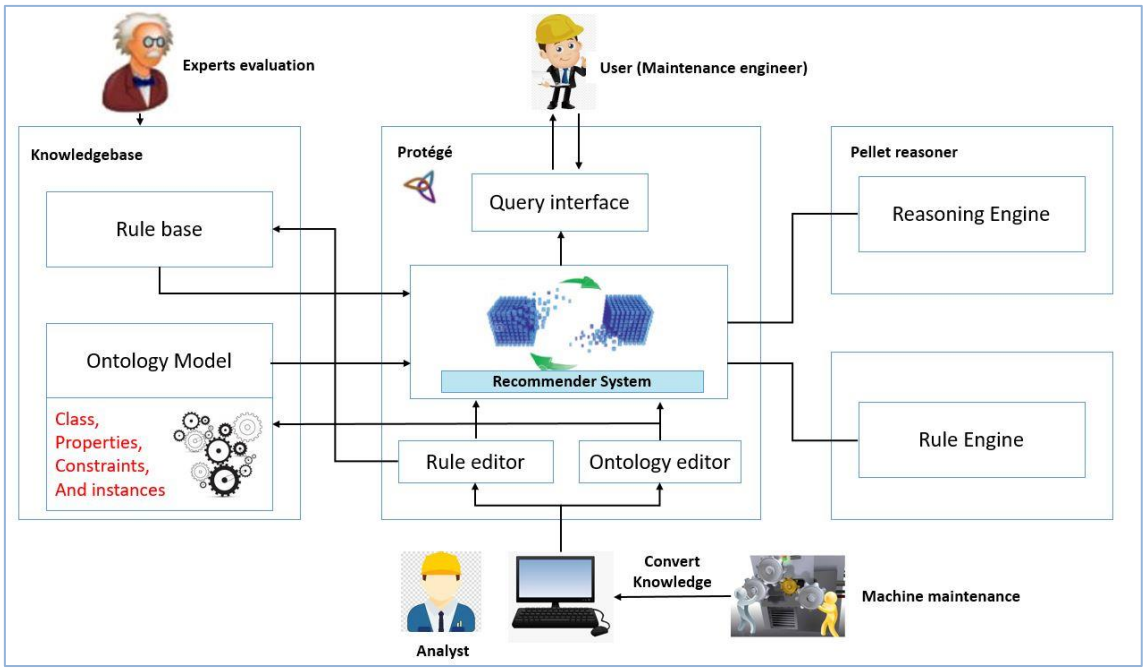


Figure 4-3: Usage viewpoint for maintenance action recommendation service

The main **activity** in the presented example is that of producing a semantic organisation of data so as to drive maintenance services.

Trigger: This activity will be initiated ('**triggered**') by the role 'analyst' in an explicit way.

The maintenance ontology is being used for the storage of knowledge relevant to fault diagnosis and reliability analysis through monitoring techniques (**Task 1**). Hence, it is possible to query which type of approach should be used for condition monitoring, and in what manner (**Task 2**). Queries can thus be made about what kind of condition monitoring technique should be used, and how. Additionally, inferences can be drawn (**Task 3**), in the sense that it is possible to make a comparison between an obtained value and specific thresholds based on relevant ISO standards in order to determine whether the value can be categorised as good, satisfactory, alert or alarm. Therefore, if a recorded value is considered to be in the alert category (**Task 4**), the system can diagnose that a failure could occur, and a maintenance notification is issued for the machine indicating that intervention is required. Subsequent to the identification of an alert notification (**Task 5**), it is then necessary to connect this with diagnostic information about

the mechanical part being investigated, which will allow the failure mode and potential causes to be determined.

4.2.3 The Functional Viewpoint

The functional viewpoint concentrates on the functional elements in a system, along with their interconnections and structure, the interfaces and their interactions, and the associations and interactivity with components existing in the outside environment. According to the context-aware IoT architecture for industrial monitoring services, it is possible to decompose a platform's functionality into six superior-level functional areas–control, operations, information, analytics, simulation and application, as illustrated in **Figure 4-4**.

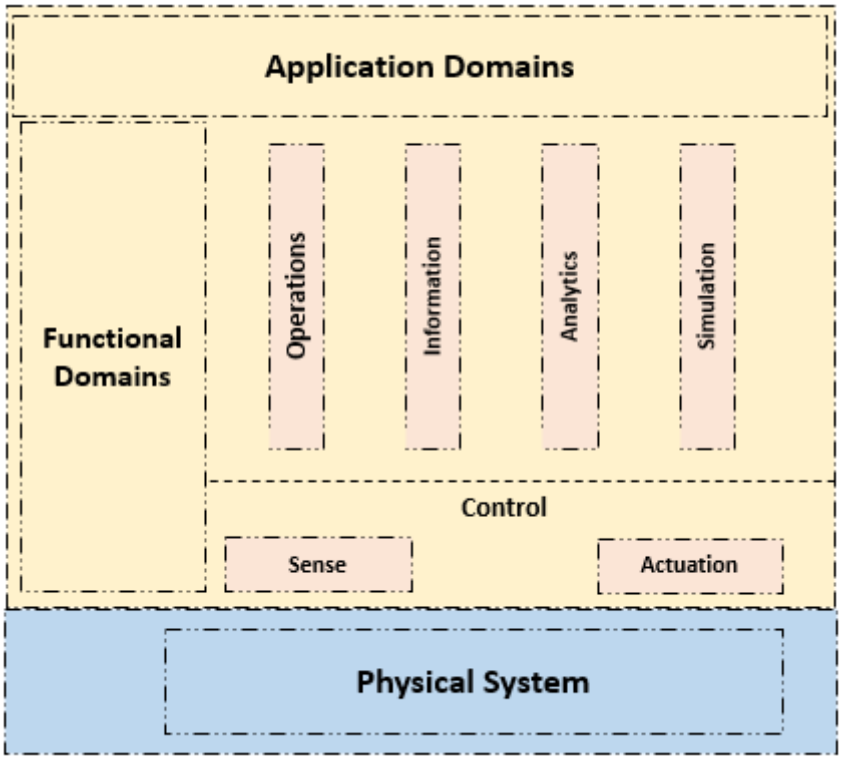


Figure 4-4: Functional viewpoint

- The control domain consists of a group of general functions. These can be implemented with different degrees of complexity and sophistication based on the specific system, and various elements may not even be included. It denotes the amalgamation of functions that are conducted by control systems in industry. The primary functions include fine-grained closed

loops, reading sensor data, the application of logic and rules, and controlling the physical system via actuators (‘actuation’).

- The operation domain denotes a group of functions tasked with providing, managing and optimising all systems within the control domain, as illustrated in **Figure 4-5**. Extant control systems in industry predominantly concentrate on the optimisation of assets within one physical plant.

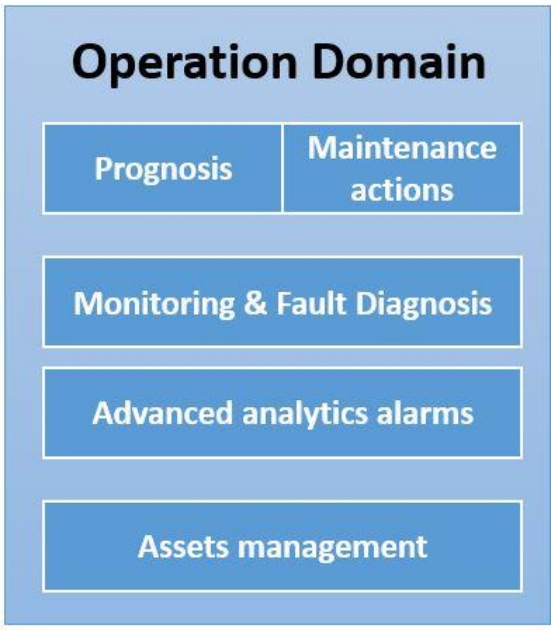


Figure 4-5: Structure of an operation domain

- The information domain denotes the group of functions responsible for collecting information from different domains as shown in **Figure 4-6**, particularly from the control domain, and then converting and modelling/analyzing the information to obtain superior-level intelligence regarding the entire system.

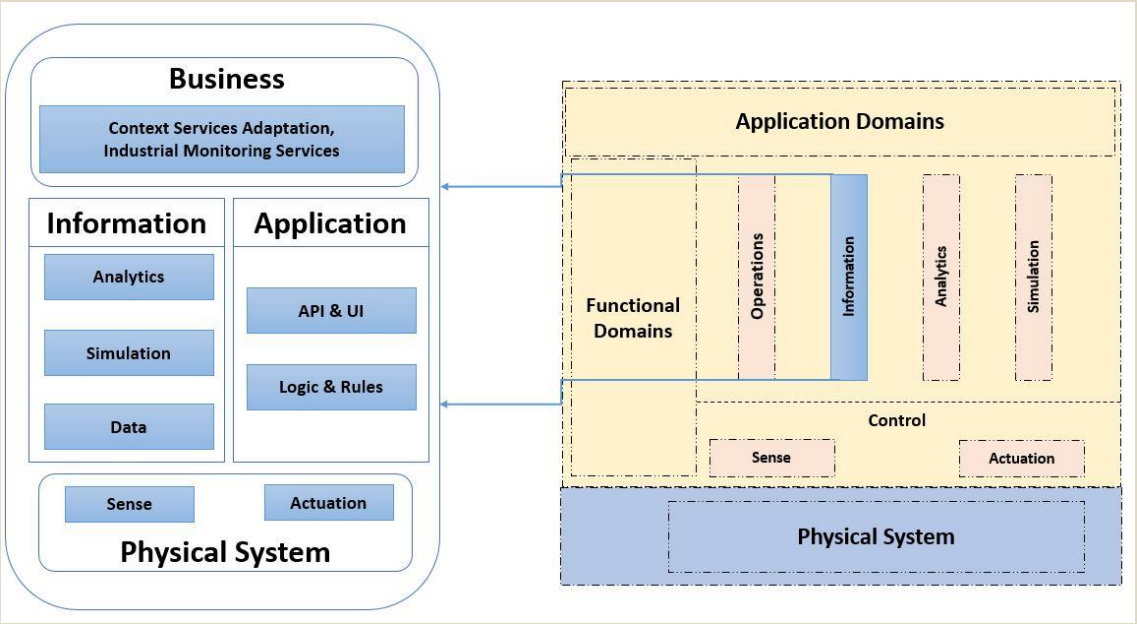


Figure 4-6: Information domain

- The analytics domain incorporates capabilities for the collection and processing of field data to facilitate the comprehension of monitoring services. It also delivers intelligence to users, although this is not always restricted to humans or vertical implementations (i.e. predictive maintenance solutions). When configured correctly, both automation and simulation domains can directly utilise the results of data analysis algorithms.
- The simulation domain incorporates capabilities for the simulation of behaviour, with the goal of optimising or verifying what-if cases incurring low costs and risks.
- The application domain denotes the group of functions responsible for the implementation of logic to realise particular business capabilities.

As IoT technologies become more embedded in monitoring activities, there is a growing necessity to manage their context information in industrial environments. Cloud computing is particularly relevant, enabling the delivery of hosted services, such as storage, networking, analytics and software development platforms, over the Internet. Hence, there is a need to ensure that context-sharing platforms offer context interoperability. Context-relevant

data can be produced by IoT entities, and context management needs to be handled through a context-management information-processing layer, which in turn is expected to handle contextual data produced from multiple sources, including third-party software. To clarify how a context-sharing platform is organised, **Figure 4-7** illustrates an example of how context sharing applies in an IoT environment for monitoring services. With asset monitoring, users are able to capture the state of a machine so they can understand how the asset is performing in the field. In this example, a physical gearbox test rig was considered as an asset, designed to emulate misalignment. The application of context sharing allows monitoring of the performance of the asset at the same time as making a diagnosis and determining the required failure modes and maintenance actions.

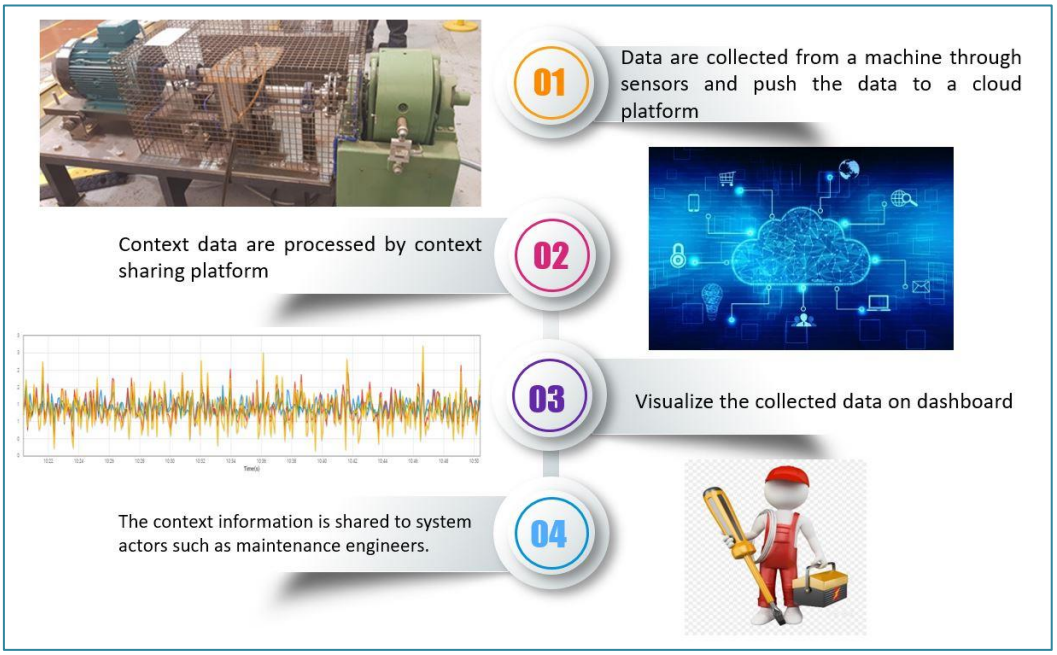


Figure 4-7: Context sharing feature

Data are collected through different sensors (i.e. temperature, pressure, proximity and humidity sensors, and accelerometers) and are fed into an embedded computing device. The device is networked, and is able to push the data to a cloud platform, where they can be processed and visualised into information relevant to the users. This information can be presented in different forms, the most common being simple monitoring of the raw data on

graphs to generate suggestions for possible interventions based on measurement ranges. When unexpected or unwanted events are detected, the system issues alerts, which are communicated to the users or other systems.

4.2.4 The Implementation Viewpoint

The implementation viewpoint addresses the technology required for the implementation of functional elements, their communication schemes and their lifecycle processes.

Figure 4-8 presents the implementation viewpoint for the integration of IoT and cloud computing for industrial monitoring services, which (i) provides a common terminology as well as (ii) mapping to extant architecture representations. It commences with a definition of each of the elements depicted in **Figure 4-8**, beginning in the lower part. To establish a clear differentiation between the notions described in this study and analogous or similar notions depicted in the relevant platforms and associated research:

- The field layer is defined as the physical layer, which is equipped with sensors designed to detect and collect information regarding the environment. The primary task of this layer is to gather beneficial data/information from objects or the environment and then convert them into a digital setup.
- Sensors are hardware employed for the measurement of parameters in their physical surroundings, which are converted into electrical signals; for instance, this could involve measurements of the humidity or temperature levels of a piece of equipment.
- An additional hardware component is the actuator, which can take actions on, control or influence the physical environment, such as by providing an acoustic or optical signal. These connected gadgets deliver commands to the actuator, and convert electrical signals into a certain form of physical activity. Similarly to sensors, actuators generally have a connection with, or are embedded in, a gadget, and this connection can be established either in a wired form or wirelessly.

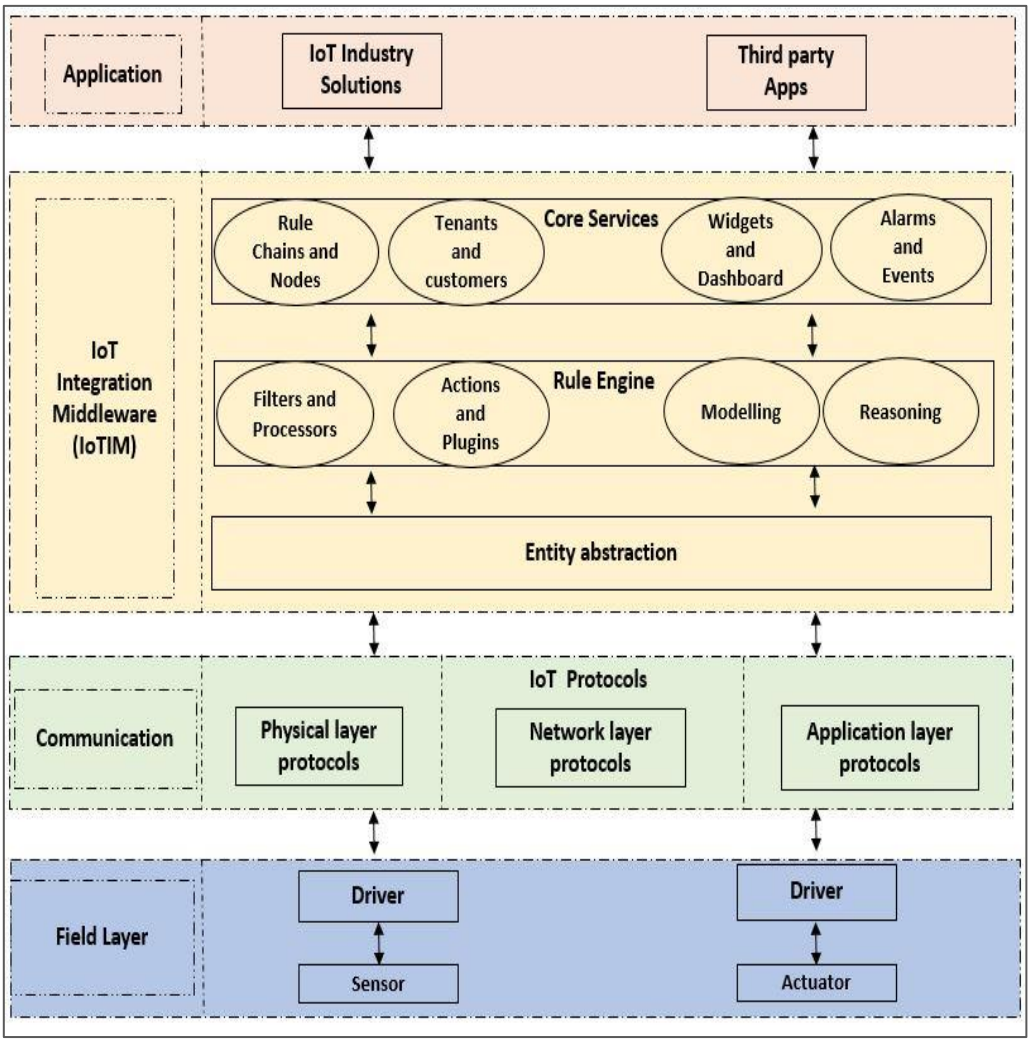


Figure 4-8: Context-aware architecture for the integration of IoT and cloud computing for industrial monitoring services.

- A device is also a hardware constituent to which sensors and/or actuators are connected, either in wired form or wirelessly, or sometimes these components are embedded inside the gadget. The processing of information received from sensors and actuator controls generally necessitate software, consisting of drivers. In the context of the architecture in the present study, a driver facilitates the process of additional software on the gadget from accessing actuators and sensors. An initial step is to use software for processing information generated by sensors, and to manage the actuators that manipulate the physical environment. Hence, devices represent the point of entry of the physical environment into the digital universe. Devices can be either: (i) self-

sufficient, or (ii) linked to a different system, such as IoT integration middleware.

- Communication refers to the connections between sensors, controllers, actuators, gateways and different types of edge systems. Various types of mechanisms of communication are available, including a bus (local to an underlying system platform or remote) or an architecture based on networks (hierarchical, hubs and spokes, meshed, point-to-point), with some having a static configuration and others being dynamic. A protocol is defined as a unique collection of rules and regulations that are used by an endpoint in a telecommunications connection when required to establish communication with a different endpoint that has a connection to the same, or an alternative, network. Different types of messaging protocols are available for selection based on different kinds of specifications related to an IoT system.
- IoT integration middleware is tasked with collecting information from the linked devices in order to commence processing the gathered information, such as by assessing condition-action rules, to deliver the collected information to linked applications, and to manage gadgets by distributing the commands to be implemented by the actuators. Gadgets can maintain direct communication with IoT integration middleware in cases where suitable communication technologies are supported, such as WiFi, a matching transport protocol, such as HTTP or MQTT, and a corresponding payload format, such as JSON or XML. Alternatively, communication is facilitated via a gateway, using IoT integration middleware. Hence, from the functional perspective, it acts as an integration layer for various types of actuators, sensors, gadgets and applications.
- Entity abstraction, by representing a virtual entity, delivers an abstraction of multiple actuators and sensors, peer controllers and systems within the succeeding higher tiers, and describes their interconnecting relationships. It acts as the context that facilitates the understanding of sensor data, the enactment of actuation and the process of interacting with different entities. In general, it incorporates the semantics of the terms used in the

representations or messages communicated among elements of the system.

- The IoT integration middleware is not constrained to the abovementioned functionalities. It can also consist of different types of functionality that are essential for particular cyber-physical systems, such as rules engines and graphical dashboards. Furthermore, the management of gadgets and users, in addition to the compilation and utilisation of collected information, can be conducted within this component.
- The rules engine facilitates the processing of messages from devices, and initiates configurable processing modules, known as plugins. Via the rules engine, it is possible to deliver an email in the event that the gadget properties change, set a notification to alert when telemetry values surpass a given threshold, and redirect telemetry information to Kafka or an external RESTful server.
- Core services include a collection of primary services that facilitate the management of certain entities, including gadgets and their associated credentials, rule nodes and chains, tenants and clients, widgets and dashboards, and alerts and events. Rules are capable of invoking a specific subgroup of APIs; for instance, a rule could set an alert for a specific gadget.
- The application component denotes software that utilises IoT integration middleware to increase the understanding of the physical environment by requesting sensor information, or to manage physical actions through the use of actuators; for instance, a software system that manages a machine's temperature is an example of an application linked to IoT integration middleware. In this context, an application could additionally be a different form of IoT integration middleware, such as for integrating numerous systems.

4.2.5 Summary

An architecture includes information that identifies the basic architecture constructs and defines the concerns, viewpoints, stakeholders, types of model, rules of correspondence and applicability conditions. Architecture frameworks can be used by system architects for discovering, describing and organising topics of interests (concerns) related to a given system. Furthermore, architecture representation can be used for clarifying, analysing and resolving such concerns.

In this section, the proposed architecture that introduces context awareness to enhance remote monitoring services was explained. It consists of four viewpoints—business, usage, functional and implementation–as shown in **Figure 4-9**. These viewpoints are concerned with the technical representation of an IoT system, and address the technologies required for the implementation of functional elements, along with their communication schemes.

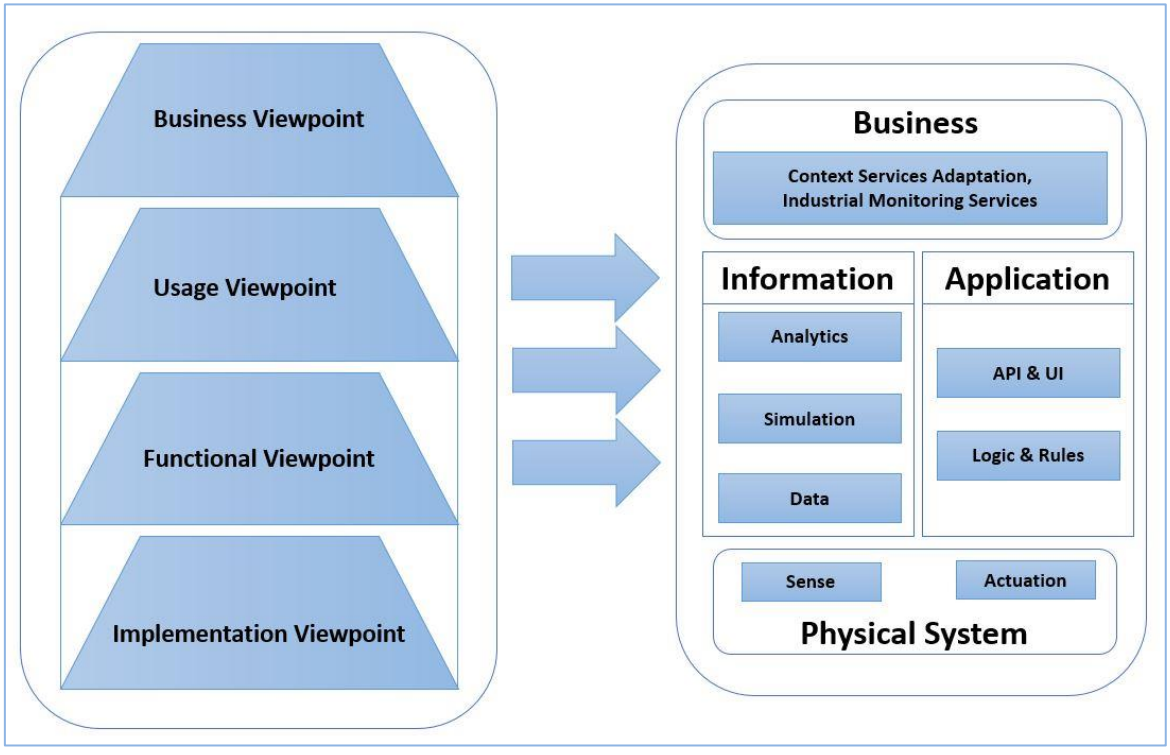


Figure 4-9: Summary of architecture viewpoints

Context information management has largely dealt with the challenges of ubiquitous environments, as well as the data heterogeneity and service

scalability. However, when dealing with monitoring services in manufacturing environments, developed approaches often lack expressiveness concerning the representation of domain knowledge. To address such needs, there is a need to develop an effective context-aware framework to enhance monitoring services in industrial environments as a means of addressing challenges related to information complexity as well as to integrate data with domain knowledge in industrial monitoring applications. The next section of this chapter will focus on a novel framework for context-aware IoT-enabled maintenance services.

4.3 FRAMEWORK DESIGN AND DEVELOPMENT

4.3.1 Introduction

When considering IoT usage in industrial environments, the term IIoT, or simply Industrial Internet, is employed, and is being considered as fundamentally linked to Industrie 4.0 (Jeschke et al., 2017). Product Lifecycle Management (PLM) systems are particularly benefitting from such technologies that connect physical assets and products, processes, data, people and business systems (Keivanpour and Ait Kadi, 2019), exploiting product-embedded sensor and intelligence capabilities, including product or process condition monitoring capabilities. However, when dealing with monitoring services in manufacturing environments, previously developed approaches often lack expressiveness concerning the representation of the domain knowledge. To address this gap, there is the need to develop an effective context-aware framework to enhance monitoring services in industrial environments as a means of addressing challenges related to information complexity, as well as to integrate data with domain knowledge in industrial monitoring applications.

One way to handle such challenges is to introduce context modelling and management, wherein data and service delivery are determined through resolving the context of a service or data request. The following section describes a framework that introduces context awareness to enhance remote monitoring services in manufacturing environments. The framework applies context-aware computing to deliver solutions and address key challenges that IoT-enabled monitoring services need to handle, specifically how the context can be modelled,

processed and disseminated for remote monitoring services, how this impacts the service discovery solution, and what an appropriate taxonomy of ontology is.

4.3.2 Overview of the context modelling and management framework

The actual value of data can be enhanced via contextualising data provisioning services; that is, by providing the right information to the right person in the right place and time to serve the needs and purposes in a particular business process. This becomes a definitive requirement when these services lead diagnostics and support decision-making by fusing information from collaborating maintenance and monitoring systems. Therefore, in order to design an appropriate framework to efficiently manage context for IoT-enabled monitoring services, it is important to separate the IoT context framework for monitoring services from the particularly frequent context (defined by Dey as the expression 'context') in computing in order to establish the background circumstances, or the particular situation of an entity regarding specific data or computing service requests. In order to do this, the composing IoT entities must initially be recognised, and their situations must be subsequently classified. It was found, following a review of current associated studies, that the information paradigm comprising entities, services and resources are necessary IoT domain actors, as depicted in **Figure 4-10**.

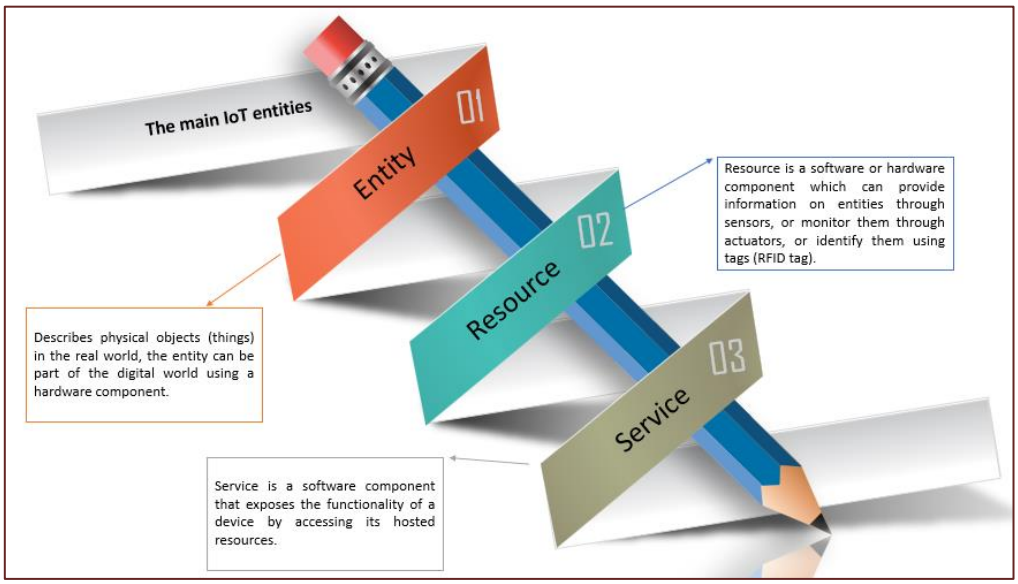


Figure 4-10: Main IoT entities

As a result, context data models have been increasingly adopted by application-focused initiatives that compete in translating maintenance mobility and context awareness into specific benefits for maintenance performance and remote monitoring. The facilitation of well-framed context information can leverage the profiling of key maintenance processes, and filter the information exchanged between integrated components.

Various researchers have thus utilised different context categorisation schemes based on their different perspectives. Abowd et al. (1999) differentiated between primary and secondary, in addition to conceptual and operational, context. It is possible to classify operational context further into sensed, static, profiled and derived, and, according to Chen and Kotz (2000), context can be categorised as passive and active, based on whether the context can be directly actioned with respect to the manner in which it is utilised in applications. Based on an operational categorisation viewpoint, Henricksen (2003) classified context into four levels—sensed, static, profiled and derived, whilst Liu et al. (2011) stated that context can be classified as user, physical or networking. Another study, published in the same year, by Yanwei et al. (2011) classified context into three levels—user, computing and things. Emmanouilidis et al. (2013) classified context into five levels–user, environment, system, social, service. Valverde et al. (2018) have focused only on the location and social. After conducting the initial experimental tests, context classes were reached that have meaningful connotations for adapting maintenance services. In this regard, **Figure 4-11** presents context categories as follows:

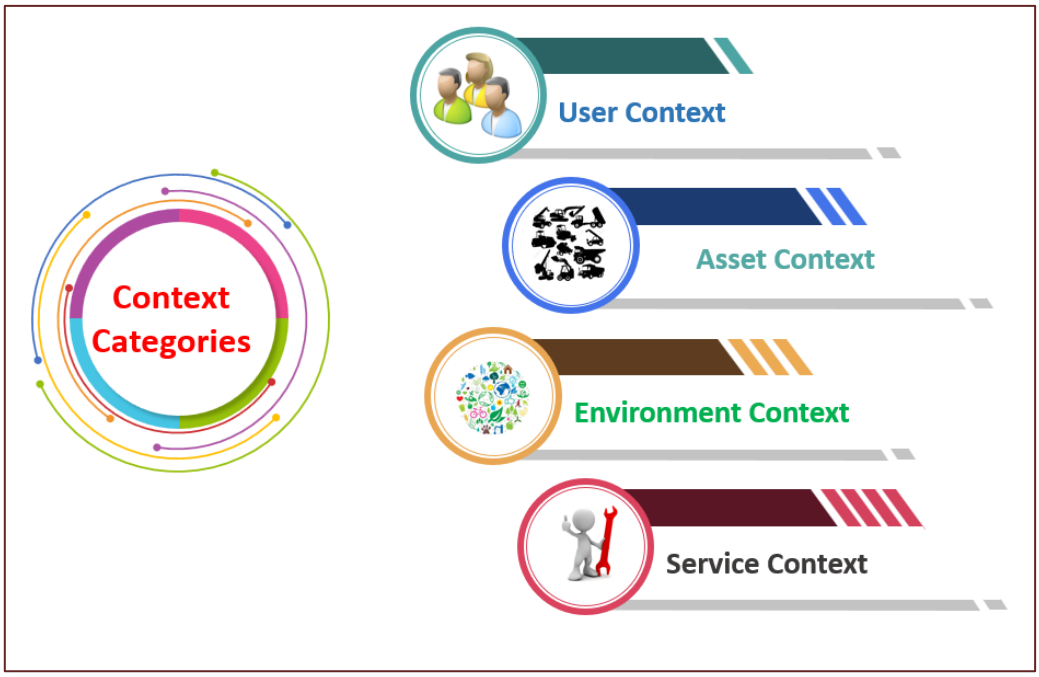


Figure 4-11: Context taxonomy for remote monitoring services based on (Emmanouilidis et al., 2019; El Kadiri et al., 2016)

- **User context** can broadly be described as the information that pertains to a user and that user's system. The semantics underpinning the data, such as role, task and location, are mapped by the user context. The system context contains the semantics used to describe the features of a device, as well as additional non-functional data.
- **Asset context** is defined as the attributes of industrial assets, describing how they connect with other components. This can comprise a part, component, device, sub-system, functional unit, system or equipment that can be separately explained and considered.
- **Environment Context:** All contexts surrounding a user are clustered by an environmental category. This contains environmental physical information, such as data collection parameters and measurement techniques, as well as non-functional data about things such as a device's battery or memory.
- **The Service Context** can be defined as characterising the service required by a user to produce a semantic organisation of their data so as

to drive maintenance services adaptation, and includes the context of the functions and services assigned by the user.

Context is commonly translated, system-wise, into a set of parameters that define a descriptive state or a profile. The environment and the asset context are often the most easily exploitable contexts for producing system adaptations that impact the user experience. The user context is a much wider context, with a large pool of custom or domain-oriented parameters that can drive the personalisation of services. Moving into the service context, the parameter semantics become more abstract and less tangible or measurable. These contexts can easily be expanded using better focused semantics that, instead of supporting a state or values, hold descriptive knowledge.

Context awareness introduces the capturing, clustering and interpretation of the above contexts in order to balance and enrich the provision of content and services. While all these approaches deal with some form of context management, starting from acquisition and modelling, eventually actionable context needs to be domain-specific. In the application domain of asset and maintenance management, context strongly depends on assets and their hierarchies. Unless such context is captured, it is hard to convert IoT-generated data from industrial systems into actions. Therefore, it is important to create a representation that integrates qualitative and quantitative data, wherein data and service delivery is determined upon resolving the apparent context of a service or data request. The most common approach to achieve this, as presented in **Chapter 2**, is through ontology-based modelling. Context modelling is where all entities and relationships, among such entities that are required for describing the entire context, are specified; for instance, this could include location, environment or user data, as well as its current or scheduled activity. On the other hand, context reasoning denotes the automatic deduction of additional facts that were previously implicit on the basis of explicitly-provided context data.

An ontology formally represents knowledge through concepts and relationships that exist in a specific domain, and are a key construct of the semantic web. Therefore, a maintenance ontology serves as a semantic formalism that can be

employed to drive maintenance services. The mechanism for this is through resolving the context of a service request, with the context identification being performed via ontological reasoning on the basis of semantic similarity, determined by ontological distance metrics or other relevant means. Asset maintenance action recommendation cannot be considered in isolation from other operational aspects, however, and so operational semantics are of considerable value in ensuring an ontology of appropriate scope and applicability.

Ontologies have frequently been categorised according to their design and structure, considering their expressiveness and generality as key normal criteria. The generality criterion advocates the establishment of a layered view of ontologies, with its key objective being to specify general classifications at the highest levels (abstraction) and more specific classifications at the lower levels (granularity). The expressiveness criterion shows the degree of detail of an ontology. Consequently, this study concentrated on two levels of detail for context modelling–upper and lower. The objective of the upper level was to supply a fundamental taxonomy of context classifications to represent very general concepts of context, as shown in **Figure 4-12**, whereas the lower level was employed to indicate a series of detailed classifications that depended particularly on the domain.

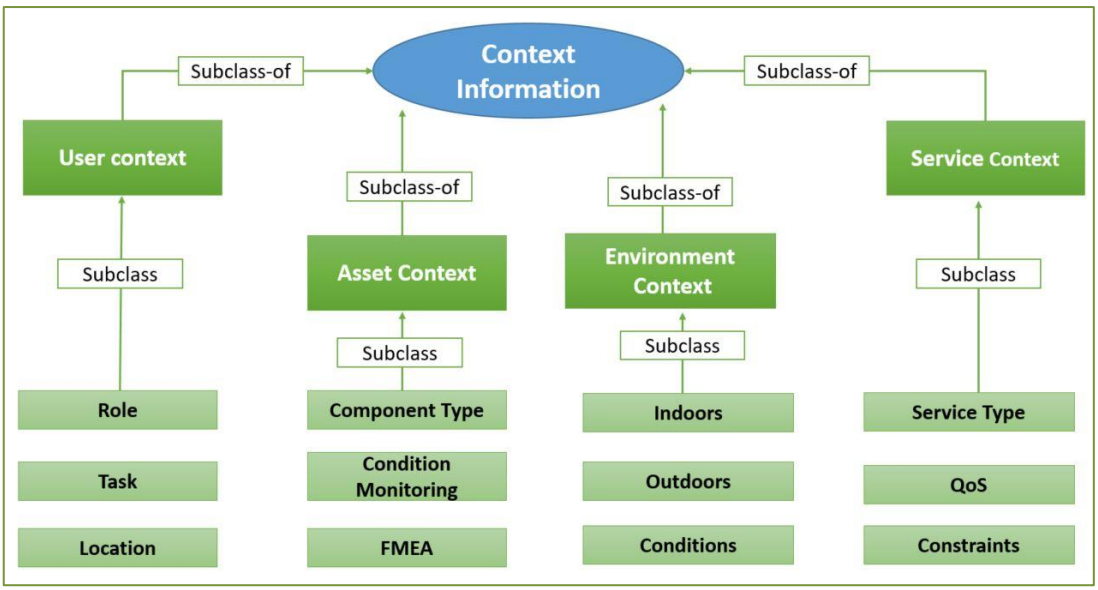


Figure 4-12: Upper-level ontology and characteristics

Focusing on the asset context, relevant domain knowledge can be modelled in many forms, but of particular interest are the knowledge constructs relevant to reliability analysis, such as FME(C)A. The asset context has to be resolved for the analysis of mechanical systems, and to establish logical connections between measurements, perceived behaviour and the desired functionality, and the operating health and defects. In this regard, FMEA provides a suitable basis for the baseline of knowledge mapping (IEC 60812, 2018), for a variety of reasons. First, the qualitative components render it suitable for the abstraction of maintenance knowledge, focused on reliability. Second, the quantitative component allows maintenance tasks to be prioritised on the basis of measurements, conducive to an approach based on risk. Third, its bottom-up structure allows failure to be assessed starting from the basic level of a production system. In other words, data are analysed from machinery parts through to the overall system. The initial stage involved the determination of the specific aspects of the machine that had the potential to fail, and then progressed to obtaining a comprehension of the causes and effects of such failures (FMEA).

Based on the modelled data, more abundant contextual information can be deduced via user-defined reasoning, allowing new knowledge and comprehension to be acquired based on the particular context. When considering the maintenance domain, real data collection from the shop floor, or simulated data with ‘has current value’ data properties, can be used to infer a component’s health status, and trigger alerts for decision-making, such as the prognosis of a failure and the scheduling of condition-based maintenance actions. In this regard, the SWRL language has been used in object properties to construct transitive rules, with new connections being applied to the classes that allow assertion inferences to be improved. In this way, the ontological approach becomes scalable. Specifically, SWRL built-ins (SWRLb) allow further extensions within a taxonomy. This greatly enhances the model by allowing multiple arguments, according to specific real-world requirements, that enable greater expressiveness in OWL2 languages.

A transitive property is considered in cases such as the following: if subclass component type (C1) has object property has mode, and subclass failure mode (FM) has object property has cause (CA) related to subclass potential cause (PC), then subclass component type (C1) has the object property has cause (CA) related to subclass potential cause (PC). Then the SWRL rule is: has mode (?C1, ?FM1) failure mode (?FM1) component type (C1) potential cause (?PC1) has cause (?FM1, ?PC1) –> has cause (?C1, ?PC1). **Figure 4-13** illustrates this rule.

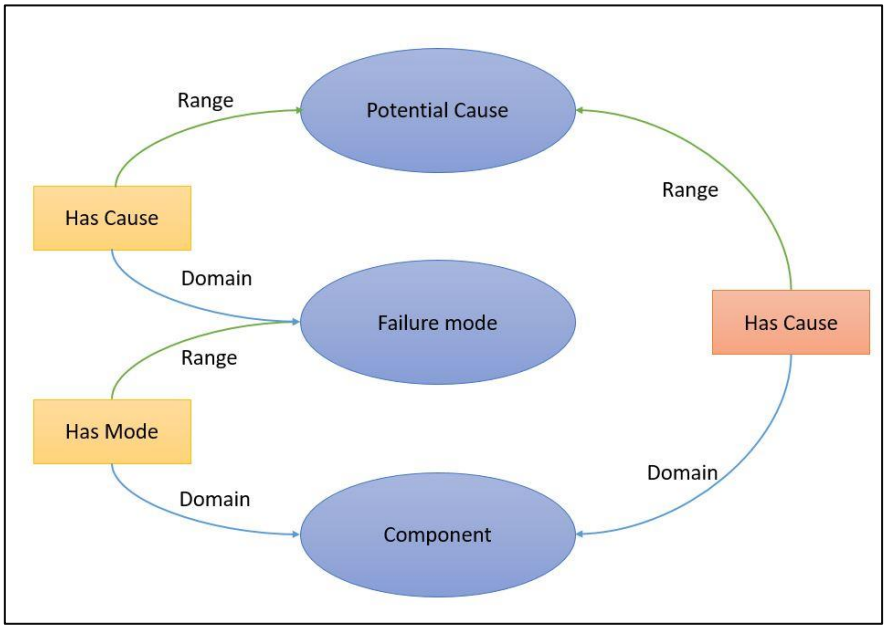


Figure 4-13: Representation of SWRL transitive rule based on (Nuñez and Borsato, 2018)

The main conclusion that can be drawn from this section is that it is important to communicate the data from different field devices, such as IoT devices and sensors, to a higher level. However, the occurrences are changed into the information presented by means of this ontology at this level. Specifically, the context modelling technique, which resolves the difficulty of representing context information, is a significant factor. It also resolves the way information is controlled for the purpose of obtaining high-level contextual information from low-level data objects, consequently enriching the domain knowledge. Therefore, a system that is context-aware must be capable of reasoning with regard to the information, and should offer a suitable prognosis on the context of the entities. In order to become context aware, it is necessary for a system to possess a

suitable context model and reasoning technique. The next section describes a framework that introduces context awareness to enhance remote monitoring services in manufacturing environments.

4.3.3 Three-Layered Context-Aware System Framework

The ability to share context among different applications is a critical necessity for the IoT, making data shared between heterogeneous systems reusable in multiple applications (Ramachandran and Krishnamachari, 2019). Context information management has been recognised as a challenge for relevant research and early on Bernados et al. (2008) developed a data fusion framework for context-awareness systems that included the following stages: (i) Obtaining context, (ii) Information picture, (iii) reasoning and decision-making. The proposed framework was also lacking with respect to context modelling, context information representation and the distribution of associated services. Perttunen et al. (2009) have surveyed popular context reasoning and representation techniques and provided an overview of the requirements for context representation, arguing that such requirements were insufficiently covered in the literature regarding the interplay between efficiency, expressiveness, soundness, and completeness, with ontology-based approaches achieved improved scalability and reuse compared to other approaches. This finding is consistent with that of Bettini et al. (2010), although scalability of on-line reasoning with a large number of entities is raised as a significant challenge. This is the case when dealing with data of significant complexity and scale, as typically encountered in IoT applications Matos et al. (2020), making it important that the semantics of IoT data are captured by appropriate context modelling to gain valuable insights (Perera et al., 2014).

Therefore, context information management has largely dealt with the challenges of ubiquitous environments, as well as data heterogeneity and service scalability. Nonetheless, while substantial research efforts have been devoted to context information management in web-based, mobile and ubiquitous computing, including IoT-enabled computing, the translation of these advances into concrete improvements in industrial monitoring services has received little attention.

Moreover, the most applicable context-modelling techniques that have been surveyed are ontology-based. However, the studied approaches lack some expressiveness concerning the knowledge representation for monitoring services in manufacturing environments. To address these needs, it is imperative that an effective context-aware framework is developed in order to enhance monitoring services in industrial environments as a means of addressing challenges related to information complexity, as well as to integrate data with domain knowledge in industrial monitoring applications.

Building upon the understanding obtained from the extensive literature review, the second RO was meant to design and develop a framework in order to accomplish this RO, the multi-phase research methodology adopted helped to facilitate the development of a context-aware framework to enhance monitoring services in industrial environments. In this respect, the knowledge and understanding gained from the extensive literature review and the experiments (early prototype and virtual prototype) was instrumental in the development of the framework.

The proposed framework was aimed at introducing context-awareness to enhance remote monitoring services in manufacturing environments. Specifically, the aim was to produce a semantic organisation of data so as to drive maintenance services. The developed framework consists of three layers, as shown in **Figure 4-14**. The framework was designed graphically, with two main elements–the title of each level (on the left) and the different functions in each level (in the centre)–and comprises three layers (from down to up)–edge, context information management and application.

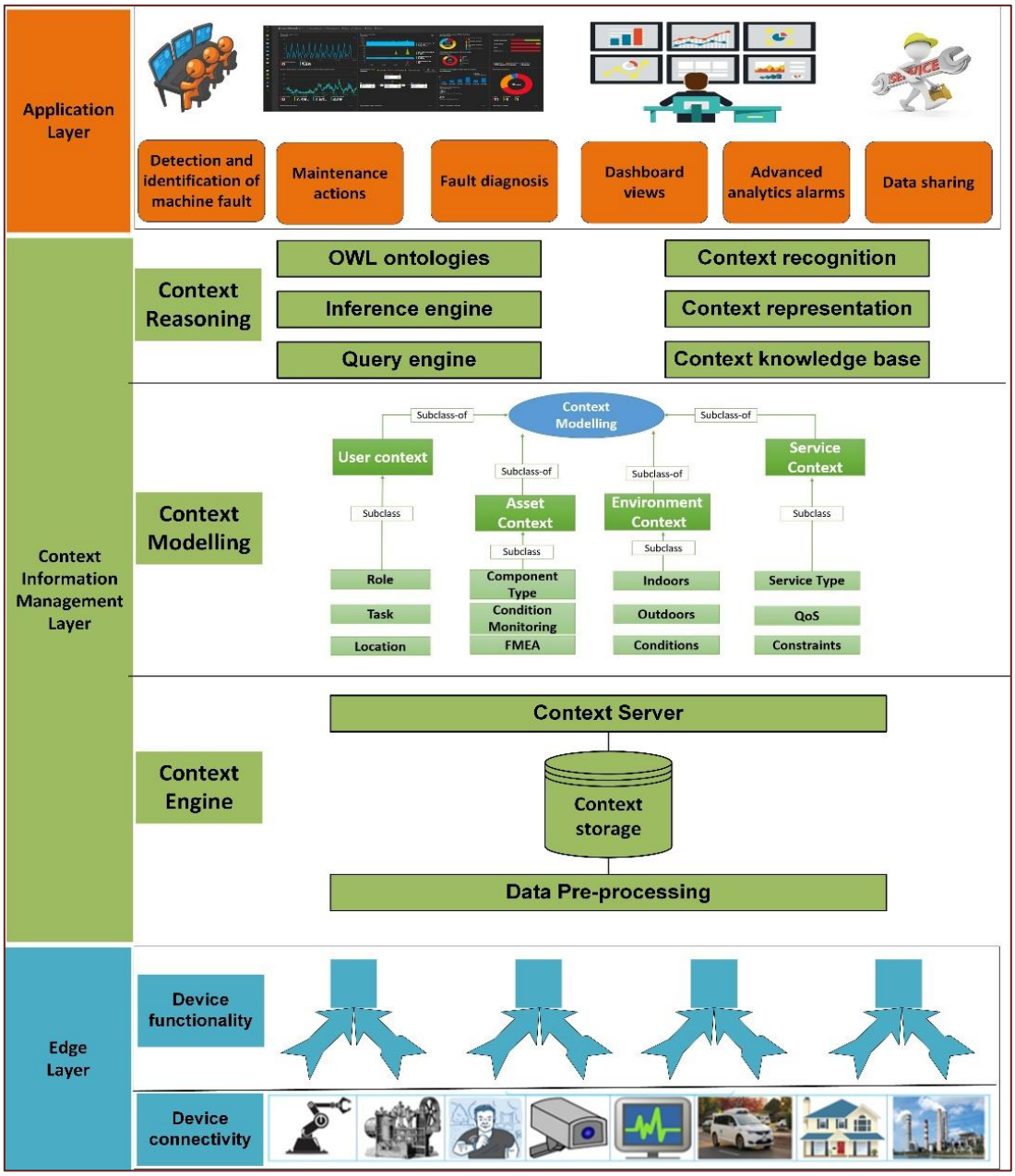


Figure 4-14: A context-aware framework for remote monitoring services

- The **edge layer** involves the sensing and collection of data via heterogeneous sensors and embedded devices. It is a combination of physical hardware and components of the network, and is tasked with the acquisition of raw context data from a variety of sources.

  There are two core modules contained within this layer device connectivity and device functionality. The former addresses the discovery and association of devices and sensors, so that they can be accessed by the platform.

Sensors and devices generally have the ability to connect via various different protocols. After discovery, a handle to the device is then delivered to the device functionality module. The handles delivered by the device connectivity module are then taken by the device functionality module and converted into high-level APIs that act independently from the low-level details of the devices and sensors. Subsequently, the contexts obtained, such as temperatures captured by embedded sensors, are passed to the context information management layer.

- The **context information management layer** pre-processes the acquired data delivered by the context-sensing layer, and then query the context ontology and infers the reasoned context. This layer functions on low-level context-sensing data so as to enable context abstraction for manipulating context at higher levels. It comprises three components that collaborate: (1) **context engine,** data obtained from the edge layer is pre-processed by the data pre-processing component to construct a context knowledge base; (2) maintenance-service-related contexts, derived from heterogeneous sources, are abstracted by the context ontology model, and then converted for the purpose of formalising representations to allow interoperability and reusability; and (3) logic reasoning services (e.g. the derivation of high-level context from contexts in lower levels) are provided by the context reasoner for pre-processing contexts pertaining to maintenance services through the reasoning engine. Below, a description of these components' functionalities is offered:
  - In principle, context-computing tasks may be delegated to a software component called **context engine**. Typical tasks of a context engine include filtering and refining the contextual clues, providing logical context interpretation, accessing external context providing services, and managing an archived sensor information database. A description of these components' functionalities is offered below: **Data pre-processing component:** Generally, it is necessary to have multiple sensors functioning at the same time for the correct identification of various activities, with each sensor delivering measurements (i.e. raw data) in a

distinct format. Hence, raw data alone have minimal (or zero) benefit for algorithms that detect activity. It is necessary for this data to be pre-processed if information that is beneficial and important for the application is to be obtained. The pre-processing stage incorporates techniques associated with cleaning, transforming, segmenting and reducing the data, with the aim of converting the data into suitable formats.

- **Context modelling (maintenance ontology**): Semantic capabilities are provided by this component through the utilisation of ontology engineering for the purpose of modelling context information related to maintenance services. In this work, the focus of the maintenance ontology was on modelling the failure analysis of mechanical components to answer queries regarding how faults manifest themselves and how they can be prevented or addressed, so as to adapt relevant diagnostics or maintenance actions in a condition-based maintenance setting.
- **Context reasoning:** The context reasoner is constructed on the context representation in addition to the functionalities of the context ontology mode. It has the objective of extracting and defining abstract context information that devices and sensors are unable to detect directly (e.g. what are the common failures and diagnostic approaches for a given machine type?; which physical parameters need to be measured/used?; what is the recommended preventive or corrective action required for the specific failure mode of an asset?), for the purpose of providing advanced processing services. This is accomplished via the use of a reasoning engine, which supports the process of inferring high-level contexts from their matching basic contexts, in a procedure that is dependent on the availability of obtained data in addition to the requirements of the system. **Chapter 5** is devoted to a detailed description of the context maintenance modelling and reasoning.

❖ The **application layer** is considered to be a top layer in the proposed framework, acting as an interface that facilitates user access to the context information delivered by the context management layer. This layer derives benefit from the various levels of contextual information that come from the

context-sensing and context-management layers. It is tasked with the delivery of application-specific services to the users. Each application may require different services, as they are dependent on the data gathered by the sensors.

4.4 Chapter Summary

The aim of this chapter was to explain the development of an architecture and framework that would introduce context awareness to enhance monitoring services in manufacturing environments. This equates to phase two of the multi-phase research methodology presented in Section 3.7 and RO 2.

Regarding the architecture development, Section 4.2 presented the proposed architecture that could introduce context awareness to enhance remote monitoring services. This consisted of four viewpoints–business, usage, functional and implementation. The business viewpoint determines the manner in which the IoT system accomplishes the established goals via its mapping to basic system functionalities. The usage viewpoint deals with the concerns of anticipated system utilisation, and is usually denoted by series of activities incorporating human or logical users, which provide its predetermined functionality in eventually accomplishing its basic system capacities. The functional viewpoint concentrates on the functional elements within a system, their interconnectedness and formation, the interfaces and interrelations, and the associations and interconnections of the system with components in the external environment. Finally, the implementation viewpoint focuses on the technical depiction of an IoT system, and the technology and system constituents necessary for the implementation of functions and activities determined by the functional and usage viewpoints.

Regarding the framework development, Section 4.3 included a three-layered framework–edge, context information management and application–that introduced context awareness to enhance remote monitoring services in manufacturing environments. The edge layer involves the sensing and collection

of context via heterogeneous sensors and integrated devices. The context information management layer includes the pre-processing of data deriving from the context-sensing layer, the interpretation of these data by the context ontology model, and the inferences drawn by the context reasoner. Finally, the interface represented by the application layer enables users to access the context information provided by the context management layer.

The development of this framework and architecture are a significant part of this study. As identified in Section 2.6, a comprehensive framework and architecture for introducing context awareness to enhance remote monitoring services in manufacturing environments has been lacking.

The next chapter explains the development of a maintenance context ontology for the framework, focusing on modelling the failure analysis of mechanical components so as to drive monitoring services adaptation. The proposed ontology for the context-resolution mechanism is relevant to the failure analysis of mechanical components, and the terminology and relationships between concepts are structured on the basis of relevant standards with a reliability-oriented knowledge grounding.

5 MAINTENANCE ONTOLOGY DEVELOPMENT

5.1 Introduction

This chapter presents a maintenance context ontology for the framework focused on failure analysis of mechanical components so as to drive monitoring services adaptation. The proposed ontology for the context resolution mechanism is relevant to failure analysis of mechanical components, and the terminology and relationships between concepts are structured on the basis of relevant standards with a reliability-oriented knowledge grounding. A mechanism for reasoning is being utilised for the delivery of context resolution, and the obtained context can introduce a metadata layer on data or events produced by either automation or human-driven means. An example of health management of rotating machinery is utilised to offer a basis for the domain context, but the actual upper-level ontology expressiveness is such that can apply to a range of machines by extending it through more specialised or application-specific detailed ontologies.

The ontology is being utilised for the storage of knowledge relevant to fault diagnosis and reliability analysis through monitoring techniques. Hence, it is possible to query which type of approach for condition monitoring should be used and in what manner. Thus, queries can be made about what kind of condition monitoring technique that should be used and how. Additionally, inferences can be drawn in the sense that it is possible to make a comparison between an obtained value and specific thresholds based on relevant ISO standards in order to determine whether the value can be categorised as Good, Satisfactory, Alert or Alarm. Therefore, if the recorded value is considered to be in the Alert category, the system diagnoses that a failure could occur and a maintenance notification is issued for the machine indicating that intervention is required. Subsequent to the identification of an alert notification, it is then necessary to connect it with diagnostic information of the mechanical part being investigated, which will allow the failure mode and the potential causes to be determined.

Nonetheless, such simple threshold-based rules often fail to apply in practice and in view of that the ontological approach does not seek to replace actual diagnostics techniques, which may involve far more efficient and sophisticated data processing. Instead it acts as a meta-layer of knowledge to drive services adaptation, and as such could work in synergy with other techniques of data processing and condition monitoring approaches. The intended end result is that the proposed maintenance intervention is more directed and tailored to the apparent context of a situation.

As shown in **Figure 5-1**, the system framework combined the information inferred from maintenance ontology and semantic rules to model the reasoning process. There are devices within the plant that generate raw data which can be collated within a group of domain entities from which events are produced. To generate valuable information about an area of interest, subsequent reasoning, various methods can be employed to model this context. The process was applied as follows:

Initially, data are collected through different sensors and are fed into an embedded computing device. Next, the collected data pertaining to the maintenance, evaluation and decision support of machines is translated by knowledge-engineers (analysts) based on FMEA analysis into ontology and SWRL rules. Real data collection from the shop floor can be used to infer a component’s health status and trigger alerts for decision-making, such as the prognosis of a failure and the scheduling of condition-based maintenance actions. In this regard, the rule's definition is used in object properties to construct transitive rules, and new connections are applied to the classes that allow assertion inferences to be improved. Subsequently, SWRL rules are executed by the rules engine, which produces new facts within the ontology management system. Lastly, queries could be raised in terms of the resolution of the monitoring service context to determining the failure mode and its potential causes, in addition to the relevant measurement parameters. Moreover, SWRL reasoning rules can be used for the evaluation of the data gathered; the prognosis of failure

is being performed, sending a maintenance message for intervention in the machine.

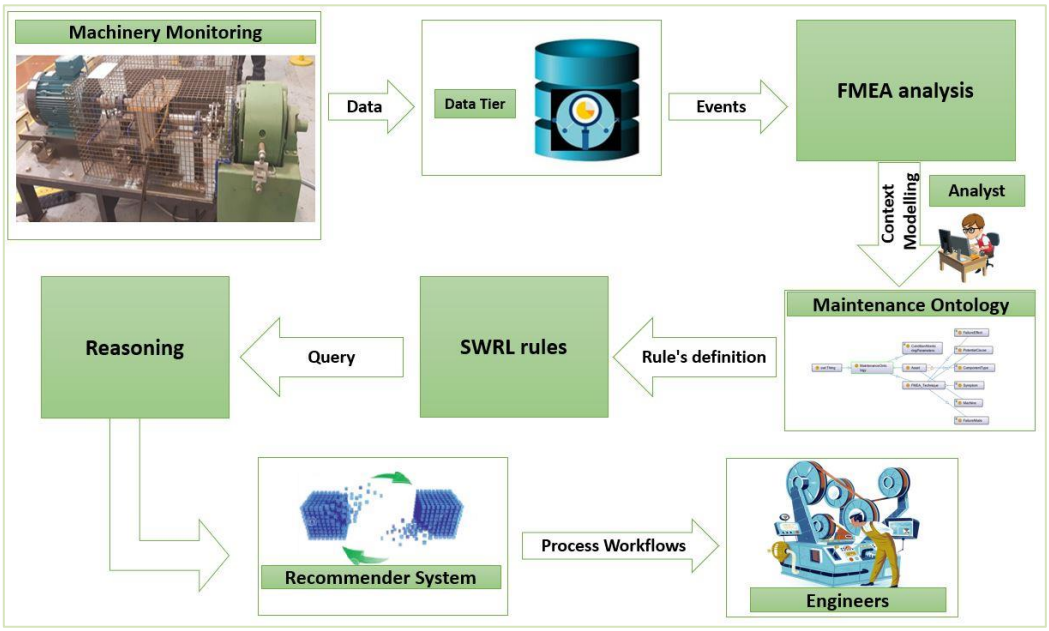


Figure 5-1: Visual representation of the system framework

A testing machine is an appropriate rig in order to emulate a wide range of failures and capture the data. In order to capture the operational health of the machine, the test rig must be analysed, allowing for an understanding on how to best capture the degradation effect on the test machine as shown in **Figure 5-2**. As stated in ISO/FDIS 17359:2002(E), which details the flowchart for starting the condition monitoring process, the start is to choose the machine components in the maintenance ontology.

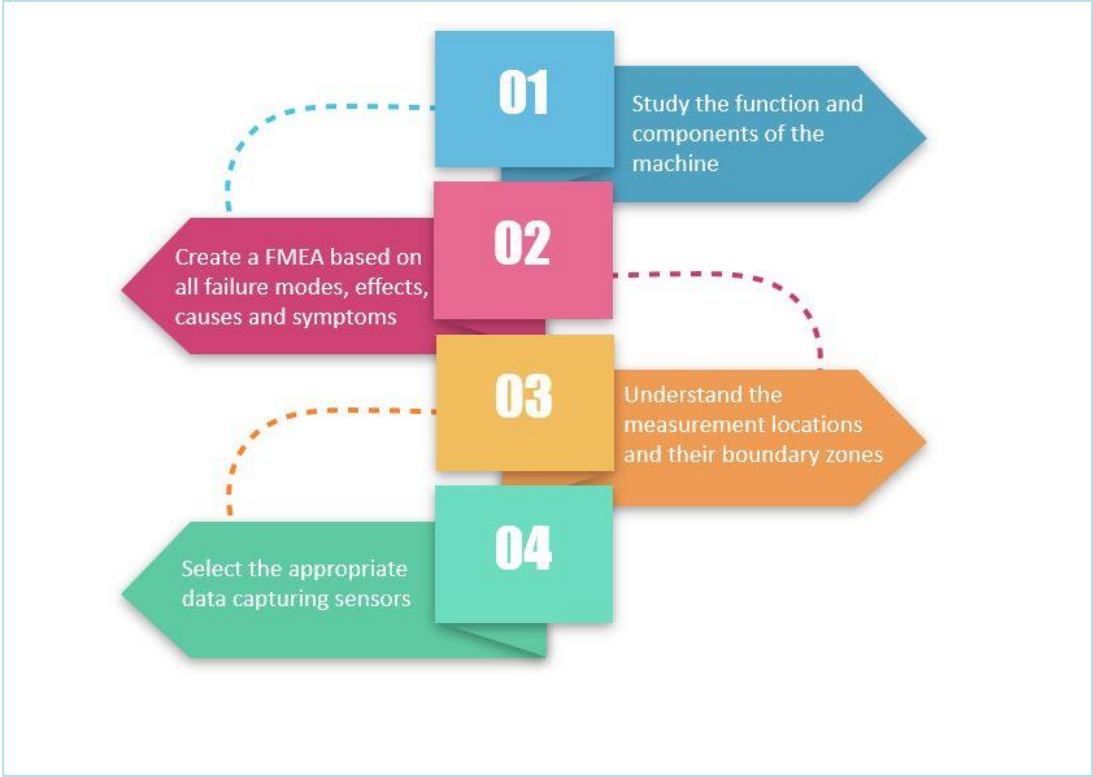


Figure 5-2: The implementation of the proposed ontology

Then, the necessary functionality of each of the components is explained for the machine to operate correctly. Additionally, all failure modes, effects, causes, symptoms and measurement approaches pertaining to the machine components are inputted utilising the FMEA method. Subsequently, the implementation of the FMEA method indicates the most suitable measurement locations and their limits for the measurement of values by employing the vibration analysis method for prediction, which are based on ISO 13373-1: 2002, ISO 13373-2: 2016 and ISO 10816-3: 2009. The most pertinent components along with the most appropriate measurement methods are identified by utilising the FMEA classification, which assigns weights according to the highest severity (SEV), occurrence (OCC), detectability (DET) and risk priority number (RPN). SEV, OCC and DET scale from 1 to 10, with higher numbers representing a greater seriousness or risk.

5.2 Failure Mode and Effect Analysis (FMEA)

Asset context must be resolved for the analysis of mechanical systems and to establish logical connections between measurements, perceived behaviour and the desired functionality, and the operating health and defects. In this regard, Fault Modes and Effects Analysis (FMEA) provides a suitable basis for the baseline of the knowledge mapping (IEC 60812, 2018) due to various reasons. First, the qualitative components render it suitable for the abstraction of maintenance knowledge focused on reliability. Second, its bottom-up structure allows failure to be assessed starting from the basic level of production systems; in other words, data are analysed from machinery parts through to the overall system. The initial stage involves the determination of the specific aspects of the machine that have the potential to fail and then to comprehend the causes and effects of such failures (FMEA).

Based on the FMEA **Table 5-1**, the most frequent outcome of misalignment of the gearbox will be vibration and power transfer loss through a gearbox, as revealed by the RPN values. Subsequently, the vibration is spread across the machine and is most pronounced in specific locations, namely the bearings, and it is possible to easily capture the transfer loss by calculating the RPN difference between the driving motor and the loading dynamometer. As determined by the FMEA analysis, the two potential failures that are identified as having the greatest level of severity if misalignment or loading occurs in the system are degradation of the gear teeth in the gearbox (RPN 150) as well as the bearing degradation (RPN 140).

Table 5-1: A subset FMEA of Test Rig based on (del Castillo et al., 2020)

Item (ID)	Function (Requirements)	Failure Mode	Failure Effects	SEV	Failure Causes	OCC	Mitigation	DET	RPN
Bearing	"To achieve a smooth, low-friction rotary motion or sliding action between two surfaces"	Abrasive wear	Reduce fatigue life and misalignment in the bearing	6	lubricant condition, grease degradation, and improper isolation	4	Lubricant inspection and proper isolation , Monitor Shaft alignment	4	96
		Bearing seizure	Crack formation on rings and balls or rollers - Skidding	4	Inadequate heat removal capability - Loss of lubricant - High temperature - Excessive speed	3		3	36
		Noisy bearing	Surface fatigue - Glazing - Micro spalling of stressed surfaces	4	Loss of lubricant - Housing bore out of round - Corrosive agents - Distorted bearing seals	3		2	24
		Fatigue (Spalling)	Bearing failure	3	Excessive loading (cyclic), misalignment	5		1	15
		Vibration	**Gear Misalignment**	7	Misalignment - Housing bore out of round - Unbalanced/excessive load	4		5	**140**
Gear	"To transmit shaft power on predetermined or designed angular velocities"	Tooth wear	Partial tooth contact **(Misalignment)**	6	Contaminants in the gear mesh area or lubrication system	5	"Lubricant inspection, Regular inspection surface sanding "	5	**150**
		Scuffing	Wear and eventual tooth failure **(Misalignment)**	5	Lubrication breakdown	2		4	40
		Tooth shear	Fracture **(Misalignment)**	6	Tooth failure	2		3	36
		Spalling	Mating surface deterioration, welding, galling, eventual tooth failure	4	Fatigue	1		2	8
		Root fillet cracking; Tooth end cracks	Surface contact fatigue and tooth failure	5	Tooth bending fatigue	2		2	20
		Pitting	Tooth surface damage	6	Cyclic contact stress transmitted through lubrication film	2		2	24

NOTE: **Table 5-1** presents a partial FMEA study (needs to be completed with domain knowledge, manually) focusing on elements of interest for the specific test rig and this as a starting ontology, which can be expanded. SEV, OCC and DET scale from 1 to 10, with higher numbers representing a greater seriousness or risk. Different combinations of SEV, OCC and DET may produce exactly the same value of RPN (have equal weights), but their hidden risk implications may be totally different. Because the RPN is the product of three ratings, different circumstances can produce similar or identical RPNs. For example, an RPN of 100 can occur when SEV = 10, OCC = 2 and DET = 5; when SEV = 1, OCC = 10 and DET = 10; when SEV = 4, OCC = 5 and DET = 5, etc. In addition, it may not be appropriate to give equal weight to the three ratings that comprise the RPN. For example, an organization may consider issues with high severity and/or high occurrence ratings to represent a higher risk than issues with high detection ratings. Therefore, basing decisions solely on the RPN (considered in isolation) may result in inefficiency and/or increased risk.

Wearing of the teeth is generally caused by misaligned gears, excessive loading and lastly, a lack of lubrication. Degradation of the bearings is caused by wearing of the teeth in the gears as well as the impact of gear vibration being transferred to the shaft and then to the bearings. If the shaft of the gear is short and hard, and the bearings are situated in close proximity to the centre where meshing of the gears occurs, this can be a source of vibration, which can be measured by placing sensors at the bearings as shown in **Figure 5-3**.

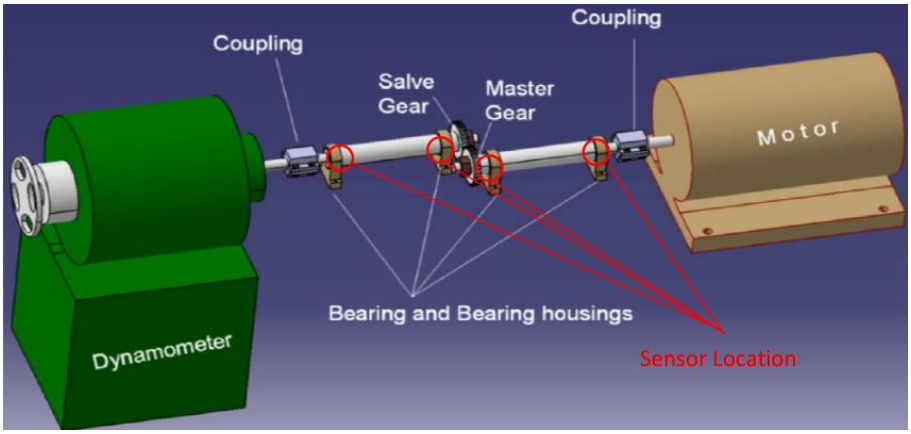


Figure 5-3: CAD Rendering of Drive System and Bearing Locations

5.3 Design of the FMEA-based ontology

The development of ontology can be based on one of the numerous procedures described in the literature (Sure et al., 2009), including Uschold and King, Grüninger and Fox, Methodology, Ontology Development 101 (OD1) and KACTUS. Each method has its own merits and shortcomings (Ren et al., 2019). In the current research, the Ontology Development 101 is adopted due to the following reasons: (1) This methodology was designed for beginners. As such, it is easy to learn and operate. (2) The detailed activities involved in this approach have been specified. The process of establishing an ontology is described in detail in this methodology. (3) It can be integrated with other tools. This method contains detailed instructions on how to implement the ontology in the Protégé environment (Ren et al., 2019). OD1 comprises six stages (Noy and McGuinness, 2001), and the way it has been applied here is shown in **Figure 5-4**. These steps are outlined next.

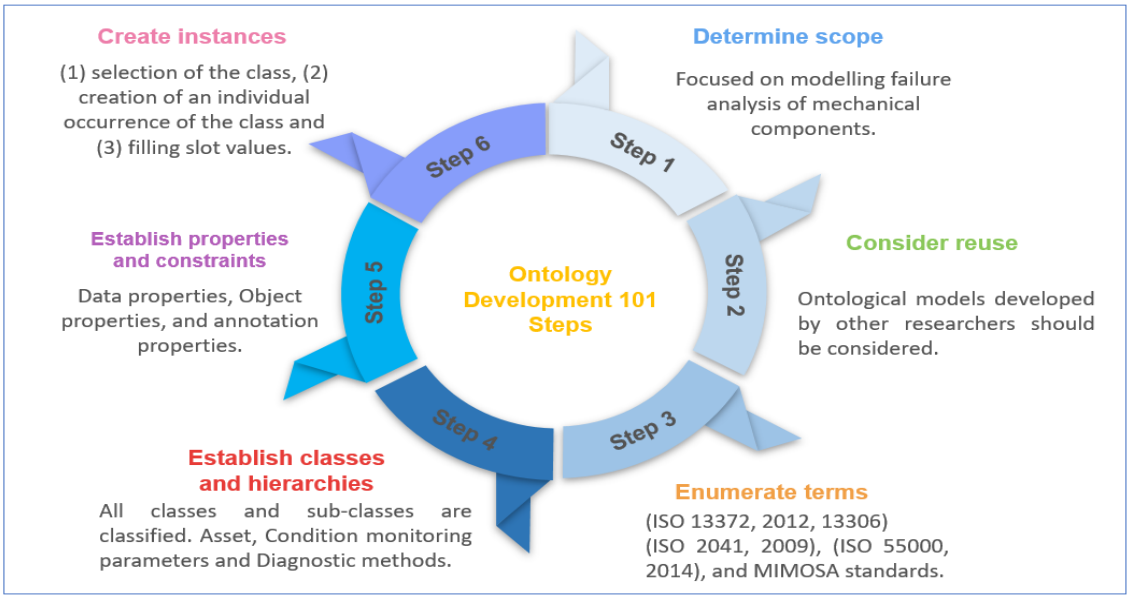


Figure 5-4: The ontology development stages

5.3.1 Determine Scope

The initial stage in the methodology is to determine the scope. It requires to define what the ontology will cover, how it will be utilised, and the types of supported questions. The responses to such questions generally evolve throughout the

process of constructing the ontology. In this work, the focus of the maintenance ontology is on modelling failure analysis of mechanical components to answer queries regarding how faults manifest themselves and how they can be prevented or addressed, so as to adapt relevant diagnostics or maintenance actions in a Condition-Based Maintenance setting.

5.3.2 Consider Reuse

The evaluation of the degree to which ontologies can be reused or expanded is a significant factor to consider. While other maintenance ontologies exist, the specific interest here is on application-specific, operational, and diagnosability concepts, thus ontological models developed by other researchers should be considered to determine their adaptability to the current research proposal such as those proposed in (Nuñez and Borsato, 2018; Sanislav and Miclea, 2015).

Nuñez and Borsato (2018) proposed an ontological model called OntoProg that served as a widely accepted data and knowledge representation scheme for diagnostic-oriented maintenance, capable of being used in different types of industrial machines. They also suggested a set of SWRL rules to improve the ontology's expressiveness. In their ontology, there is a missing link between knowledge constructs and operational and reliability-based services adaptation actions. In this regard, the OntoProg ontology was adopted and expanded for the purpose of this research. For example, the FMEA technique was adopted in this work to identify, evaluate and eliminate all potential failures or risks to a system. Moreover, the main classes were expanded. In this work, the Top-Down method was employed, in which general classes are added first, followed by the sub-classes, a process well aligned with asset hierarchies.

5.3.3 Enumerate Terms

The terminology considered for the present ontology is associated with predictive maintenance. Therefore, the main terminology and the associated definitions are based on consolidated academic literature and mostly on established international standards, such as condition monitoring, diagnostics and maintenance (ISO 13372, 2012, 13306), vibration analysis (ISO 2041, 2009),

asset management (ISO 55000, 2014), and MIMOSA (www.mimosa.org) standards.

5.3.4 Define Classes and Hierarchies

The techniques used to define class hierarchies (Uschold and Gruninger, 1996) are Top-Down; Bottom-Up; and Mixed. In this work, the Top-Down method was employed, in which general classes are added first, followed by the sub-classes, a process well aligned with asset hierarchies. The main class of the proposed ontology is **Maintenance Ontology,** and includes subclasses **Asset, FMEA Technique** and **Condition Monitoring Parameters**. Every such class has its own subclasses for example, subclass **FMEA Technique** has subclasses: **Failure Effect, Failure Mode, Potential Cause,** and **Symptom**. An example of class hierarchy is shown in **Figure 5-5**.

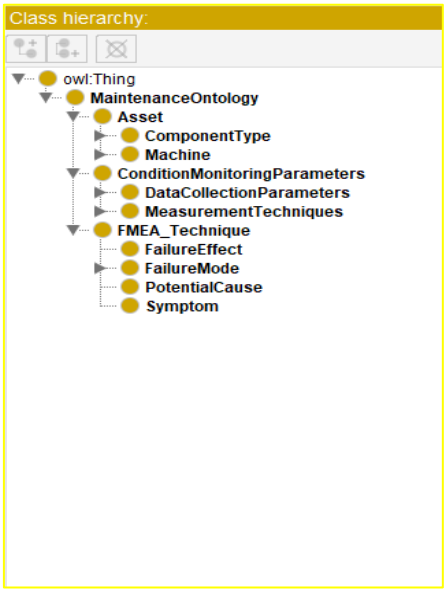


Figure 5-5: Ontology classes

A more detailed view of the first, second, and third-level classes hierarchy is shown in **Figure 5-6**, using the OntoGraf plug-in.

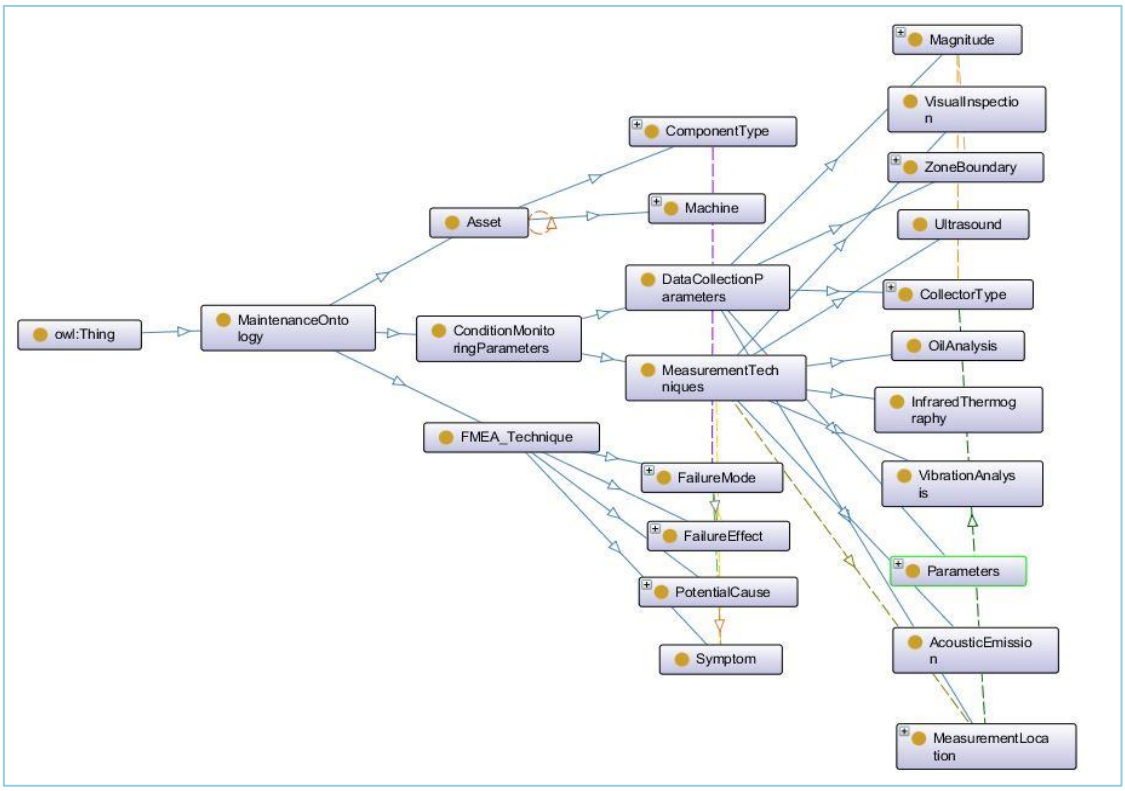


Figure 5-6: Hierarchy of level 1, 2 and 3 classes adapted from (Nuñez and Borsato, 2018)

5.3.5 Define Properties and Constraints

Class hierarchies alone are insufficient to represent knowledge. They need to be accompanied by three distinct types of properties: data properties, object properties, and annotation properties. The data property explains the properties of certain occurrences both quantitatively and quantitatively. The object attribute explains the associations among distinct classes. The annotation property is frequently employed in the description or explanation of particular occurrences. **Table 5-2** shows the properties mentioned above with their relevant Domain and Ranges.

Table 5-2: Object Properties adapted from (Nuñez and Borsato, 2018)

Object Property	Domain	Range
Has Failure Cause	Failure Mode	**Potential Cause**
Has Failure Effect	Failure Mode	**Failure Effect**

Object Property	Domain	Range
Has Measurement	Measurement Techniques	**Measurement Location**
Use Collector	Measurement Location	**Collector Type**
Use Magnitude	Collector Type	**Magnitude**
Is Part Of	Asset	**Asset**
Has Failure Mode	Component Type	**Failure Mode**
Is Detected With	Symptom	**Measurement Techniques**
Has Group	Magnitude	**Zone Boundary**
Has Symptom	Potential Cause	**Symptom**

Table 5-3 provides the terminology that gives the possibility of classes being fed with certain characteristics, called data properties, with restrictions on domain and range.

Table 5-3: Data Properties adapted from (Nuñez and Borsato, 2018)

Object Property	Domain	Range
Has Current Value	Measurement Location	**Decimal**
Has DET	Symptom	**Integer**
Has DGN	Symptom	**Integer**
Has Failure Value	Potential Cause	**Decimal**
Has Frequency Spectrum	Potential Cause	**String**
Has Function	Manufacturing Items	**String**
Has Health	Measurement Location	**String**
Has ID	Measurement Items	**String**
Has Location ID	Measurement Location	**String**
Has Measurement Direction	Potential Cause	**String**
Has SEV	Effect	**Integer**
Has Specification	Maintenance Ontology	**String**
Has Unit	Magnitude	**String**
Has Velocity	Zone Boundary	**String**
Has Warning	Measurement Location	**String**

Object Property	Domain	Range
Has Zone	Zone Boundary	**String**
Is Caused By	Potential Cause	**String**

Figure 5-7 shows an RDF graph. The mechanical component is represented by the data properties has Specification, has Function, has ID, and an object property is part of. Furthermore, this relationship enables the search for a particular component, including its description, specification, necessary function, and machine to which it belongs.

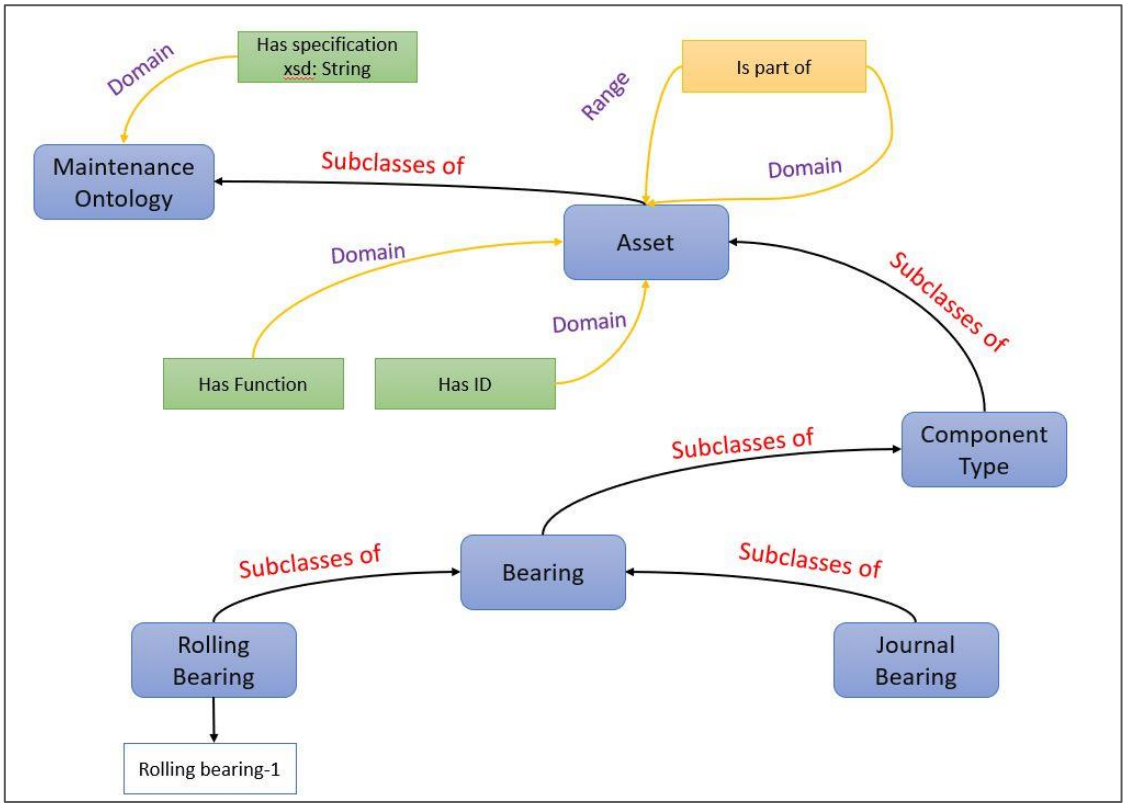


Figure 5-7: RDF graph of component characteristics adapted from (Nuñez and Borsato, 2018)

5.3.6 Create Instances

The creation of individual class instances involves: (1) selection of the class, (2) creation of an individual occurrence of the class and (3) filling slot values. These instances are used in the representation of particular elements. A class is

selected for every instance in a way that binds the properties of the object, data and/or annotations.

Along with identification of the procedure that has been adopted, the development of ontology models requires tools that can support all activities in the development process. Such tools include TopBraid (www.topquadrant.com) and OntoStudio (www.semafora-systems.com), as well as open ones, such as the popular OntoEdit (Sure et al., 2002), HOZO (Kozaki et al., 2005) and Protégé (https://protege.stanford.edu/). Specifically, Protégé is the most dominant ontology publisher due to the fact that it is an open platform that offers plug-in extensibility as well as XML (S), OWL, RDF (S) and Excel support, along with graphic taxonomy, queries in SPARQL, rules in SWRL language, and a reasoner (Pellet). The combination of OWL/SWRL provides a more flexible ontology language for modelling knowledge domains with a greater degree of expressiveness than using OWL alone (Lawan and Rakib, 2019). The SWRL is a W3C recommendation that extends horn-clause rules to OWL. OWL has demonstrated significant expressive powers over other ontology languages as the recommended ontology language for the semantic web. Although the domain knowledge models provided by OWL ontologies are not complex, they can be reused and are easily understandable, they do not have the declarative expressiveness that rules developed in SWRL can offer.

5.4 Chapter Summary

This chapter has presented a maintenance context ontology for the framework focused on failure analysis of mechanical components so as to drive monitoring services adaptation. This represented phase three of the multi-phases research methodology presented in section 3.7 and research objective 3.

It has followed an established ontology development process but its design differs from other approaches in that it expands FMEA/FMECA – based ontology constructs with additional concepts adopted from available standards in the field that link the key reliability-based concepts of the knowledge constructs with asset level and fault – specific relevant diagnosability concepts. The Ontology Development 101 (OD1) is adopted here based on available reliability standards

such as FMECA or FMEA, and has been shown to be well-suited for maintenance modelling and is well documented for implementation in Protégé environment. The Ontology Development 101 had six phases, which have been addressed as follows:

(i) Determine scope – in this phase, the scope and domain of the ontology are defined. (ii) Consider reuse – ontological models developed by other researchers should be considered to determine their adaptability to the current research proposal. (iii) Enumerate terms – this phase involves the enumeration of all terms pertinent to the area of the ontology being developed. (iv) Establish classes and hierarchies– in this phase, all classes and sub-classes are classified. (v) Establish properties and constraints– subsequent to defining the class hierarchy, it is important to determine the class relationships. (vi) Create instances– this denotes the final stage of the procedure in which specific instances are formed within the class hierarchy.

The next chapter describes the applicability of the developed ontology model by utilising a real physical asset. This is done through a case study and expert judgements.

6 CASE STUDY, RESULTS DISCUSSION, AND VALIDATION

6.1 Introduction

The applicability of the developed ontology model is shown by utilising a real physical asset. Gearboxes have broad utilisation in numerous applications such as machine tools, industrial devices, conveyors and essentially any form of rotatory power transmission equipment in which the torque and speed requirements need to be changed. When such devices fail, the results can be catastrophic, with serious consequences. Therefore, a proactive approach must be adopted that enables such components to be monitored in real time using predictive maintenance methods (Khan et al., 2019). Moreover, for the majority of rotating machines used in process industries, the preference is for faster speeds, greater horsepower per machine, and reduced sparing.

However, to reduce vibration, and prevent bearings, couplings and shaft seals from wearing out too quickly, faster speeds require greater precision with respect to balancing and alignment. Greater horsepower and less sparing substantially increase the reliability of machines, and are thus of immense economic value. These, in turn, require essential components for them to function without breaking down or prematurely wearing out. For the current study, a laboratory-based test rig was employed as a case study, as shown in **Figure 6-1**, with data being collected from its operation and maintenance records. The test was designed to emulate a complex case of misalignment, which had relevance to manufacturing and aerospace engineering assets. The technique of vibration analysis was approached because techniques used for assessing the health of components based on vibration are regarded as applicable in numerous reciprocating and rotating machines.

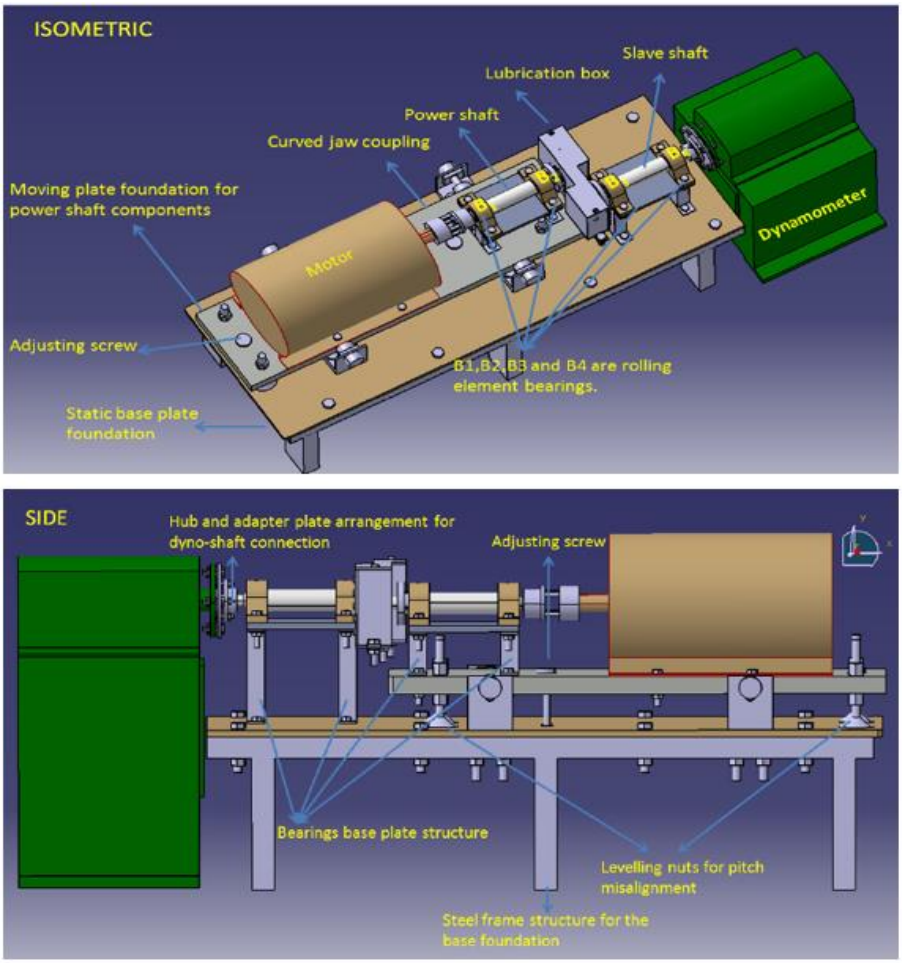


Figure 6-1: Gear-misalignment machine

6.2 Case Study Results and Discussion

The aim of ontology-based context modelling is to produce a semantic organisation of data so as to drive maintenance services adaptation. When users interact with systems in this regard, the proposed maintenance ontology can help them to narrow down their list of options to the contextually-relevant ones by providing answers to questions such as:

- What are the common failures and diagnostic approaches for a given machine type?
- Which are the physical parameters to measure/use?
- What is the recommended preventive or corrective action required for a specific failure mode of an asset?

To achieve this, context-based adaptation software was developed, as shown in **Figure 6-2**. This software represents the developed ontology, and can be used to drive maintenance service adaptation. In the simplest case, the adaptation mechanism automatically retrieves relevant knowledge content via ontology reasoning. For example, it can retrieve the most common faults expected to occur, and match the appropriate parameters or techniques suitable for monitoring the asset condition and identifying faults.

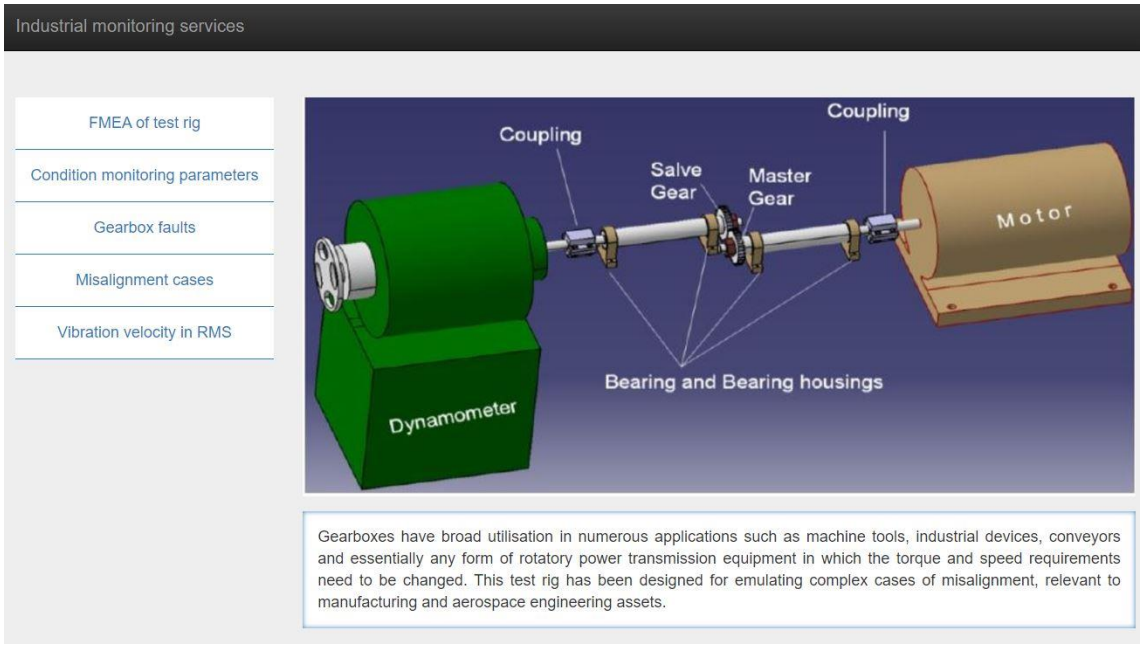


Figure 6-2: Context based adaptation interface

A typical utilisation scenario is one where, during condition-monitoring, queries could be raised to resolve the monitoring service context. For instance, this could be related to determining the failure modes of a part, the functional effect of a defect on the operation of the test rig, or the measurements suitable for identifying specific defects on specific components, along with the relevant measurement parameters. In the context of the present study, SPARQL queries were designed to resolve these queries. SPARQL also allows the federation of queries across different sources of data. By applying these queries, it is possible to determine, for example, ‘What are the main components of the test rig?’. The test rig is just an example of an asset, and the same mechanism can be applied to other physical assets.

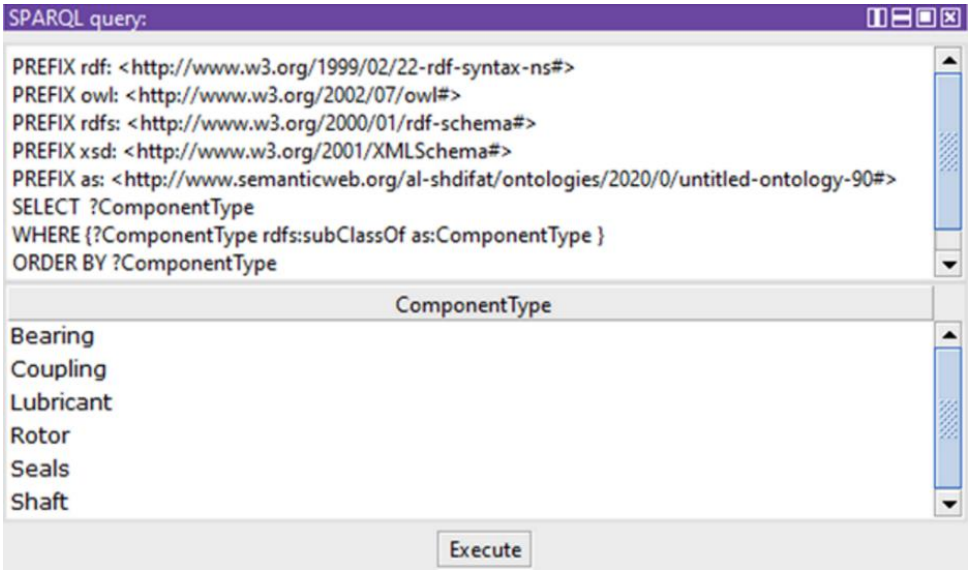


Figure 6-3: Query result to identify the main components of an asset

Figure 6-3 shows the results of a query to identify the main components of an asset type (here, a test rig), including the bearing, coupling, lubricant, rotor, seals and shaft. Another query can be applied to determine the functions of the main components. This query may be useful to a maintenance engineer, enabling the linking of faults to functional impacts, as shown in **Figure 6-4**.

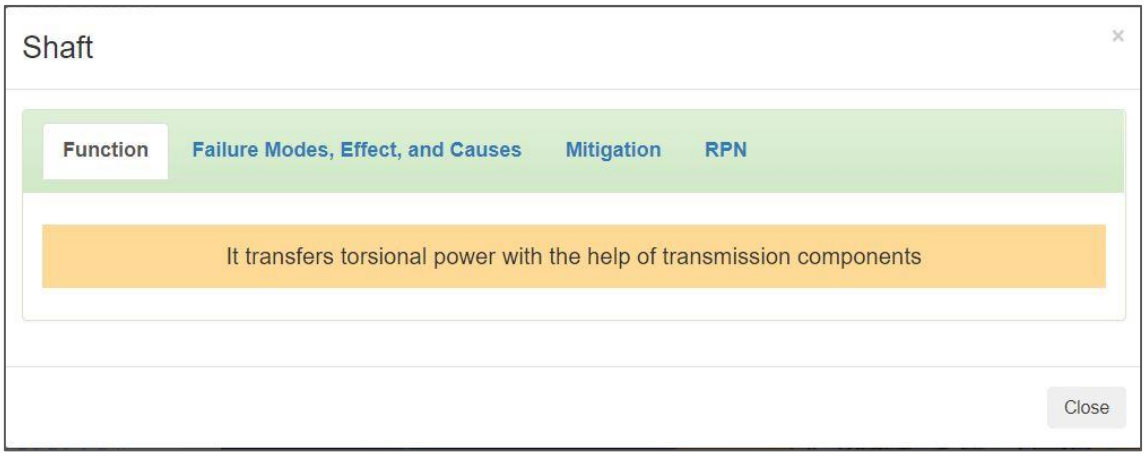


Figure 6-4: Query result to identify component functions

Figure 6-4 shows the function of a shaft component, as an example. This implementation also allows a query in the maintenance ontology to resolve key analysis characteristics, such as component functions, failure modes, causes, effects mitigation and risk priority number (RPN), as shown in **Figure 6-5.**

Shaft

Function | Failure Modes, Effect, and Causes | Mitigation | RPN

(SEV * OCC * DET) = RPN

Failure Mode	Failure Effects	Failure Causes	SEV	OCC	DET	RPN
Bent shaft	Assembly vibration - Damaged bearing, impeller, wear ring, mechanical seal, gear box	Excessive load, torque, Impact loads, Bearing failure	4	5	4	80
Excessive shaft deflection	Eventual shaft damage - Damaged bearing, impeller, wear ring, mechanical seal, gear box	Dynamic loading on shaft - Reversing loads - Critical shaft speed exceeded - Unbalanced load	3	4	3	36
Shaft misalignment	Assembly vibration - Damaged bearing, impeller, wear ring, mechanical seal, gear box	Improper assembly - Worn bearings - Excessive load	3	2	3	18
Damaged surface finish	Shaft bearing failure	Surface cracks, eventual shaft failure - Bearing, gear, coupling corrosion	2	2	2	8
Shaft fatigue / fracture	Stress riser at fillet - Stress concentration at keyway - Shaft radii changes - Bending fatigue - Excessive velocity - High torque load	Damaged bearing, impeller, wear ring, mechanical seal, gear box	5	3	4	60
Fretting corrosion	Relative movement of tightly fitted parts	Surface cracks, eventual shaft failure - Bearing, gear, coupling corrosion	5	3	4	60

Figure 6-5: Characteristics of shaft key analysis

Another query can be applied, based on FMEA, to determine the common gearbox problems and diagnosis methods; for example, the problems that arise in relation to gearboxes are misalignment, bearing damage, bearing wear, unbalance, mounting fault and damaged impeller, as indicated in **Figure 6-6** (left side).

Figure 6-6: Misalignment faults matched to measurement parameters and techniques

After identifying the common gearbox problems, as shown in **Figure 6-6** (left side), the parameters that can be used in fault detection can be identified (these are the parameters relevant to the specific context, given the asset type and failure mode). The developed ontology links physical measurement entities with appropriate measurement techniques. This allows the association of common faults with the physical asset, and matching them with the parameters or techniques appropriate for detecting the occurrence of such faults (**Figure 6-6**, right side).

Considering that the test rig used in this study was designed to examine complex cases of misalignment in industrial machinery, the focus here was on misalignment. In gearbox arrangements, misalignment can cause gear and bearing pitting, which eventually leads to complete failure. It may cause vibrations and excessive loads that harm the functioning components of the machine, such as bearings and oil seals. It is therefore important to detect and fix such issues to avoid incurring unnecessary costs. As shown in **Figure 6-7**, this can take four forms–axial misalignment, offset or parallel misalignment (where the centres of the shafts are in different lines), angular misalignment (where a motor shaft is at an angle to a driven component shaft), and combination misalignment (where both angular misalignment and parallel misalignment co-occur).

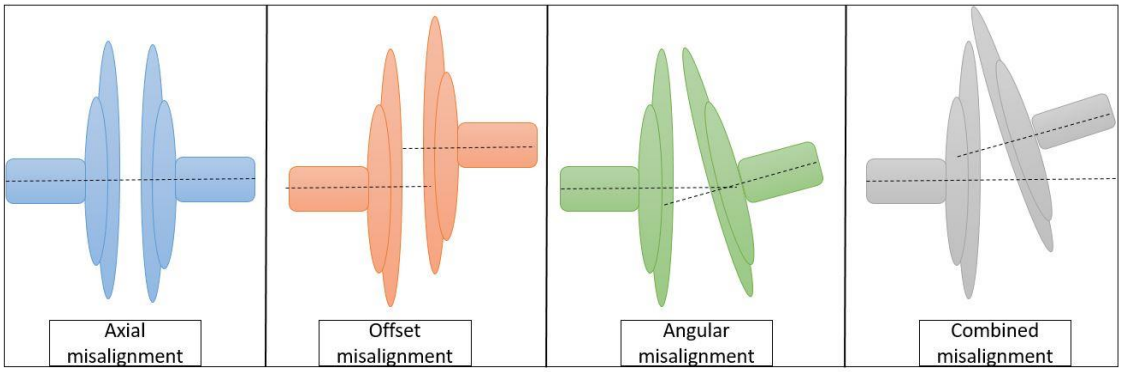


Figure 6-7: Types of shaft misalignment (adapted from Khan et al., 2019)

Components that are misaligned also consume more power. Coupling and bearings temperatures can also increase, which is problematic because the components can only function within specified temperature ranges, resulting in the breakdown of bearings, couplings and seals. Misalignment can also increase the bearing load, affecting life span and reliability.

In addition, the shaft loading caused by misalignment wears away the seals. These have precisely designed components, with an established accuracy of 2 µm, and consequently will not function well in conditions of poor alignment and increased temperatures. They can fail catastrophically, without warning. This may then cause the bearings to fail because non-functioning seals allow dirt, grit and metallic particles to contaminate the bearings. However, removing and refitting seals can also damage the bearings. Both must therefore be replaced. An additional problem caused by misalignment is that it generates reaction forces in the coupling, which often causes vibration. The primary purpose of flexible couplings is to connect shafts. They generate driving torque, and accommodate any intended misalignments. However, fixing of the coupling moments can lead to the machine shafts bowing. This worsens as the torque and speed of the machinery increases, leading to greater vibrations.

Techniques for monitoring vibration and noise that may affect health are considered to be a plausible option for a large number of reciprocating and rotating machines, and gearboxes in particular. In the case study, optimal mesh alignment of the gears was required to ensure power was being transmitted efficiently. This ensured a uniform distribution of load on the teeth, which

extended the service life of the gears and diminished the likelihood of them failing prematurely. One of the techniques used to monitor and diagnose misalignment is signal processing, which utilises vibration analysis. This is a frequency domain analysis that uses fast Fourier transform (FFT). The FFT spectrum provides measurements in both radial and axial directions for the drive and non-drive parts of the motor. Velocity in root mean square (RMS) is measured using an FFT analyser, in conjunction with an accelerometer. The velocity frequency spectrum has a frequency that equates to rotor speed. If the harmonics of a frequency are high, there is a problem with angular misalignment. If vibrations become increasingly varied, there is a misalignment issue. Therefore, vibrational analysis can provide a fast and accurate assessment of the state of a mechanical system.

One component that has great significance in failure analysis is the bearing (**Table 5-1**). A critical failure mode is gear tooth wear, and the typical failure effect of this is partial tooth contact (misalignment). A query can be applied to determine the failure mode, failure cause, failure effect, symptoms and fault severity (SEV), and also to determine faults with the highest diagnostic potential (DGN), or those that pose the highest impact risk. SEV and DGN scale from 1 to 10, with higher numbers representing a greater seriousness or risk; an appropriate query can return the faults with the highest DGNs (**Table 6-1**) or risks. Therefore, parameters such as SEV, DGN and DET, from the FMEA technique, can be used in the ontology model to enable queries that, in turn, can identify components or processes of priority for maintenance actions.

Table 6-1: Query outcome for failure mode with highest DGN

Component Type	Failure Mode	Failure Cause	Failure Effect	Symptom	SEV	DGN	Technique
Rolling bearing	Tooth wear	Tooth failure	Partial tooth contact (misalignment)	Vibration 5	6	5	Vibration analysis

As such, the most suitable points to capture the degradation of the system, including the wearing of teeth and bearings due to loading and misalignment, are the bearings, where vibration sensors can be placed (Girdhar and Scheffer, 2004). These sensors are able to pick up the constant vibration occurring in the

system (datum vibration). Then, once the machine experiences an induced failure, the system is able to capture the variation in data, and send an alert about a potential issue in the system (Farrar and Worden, 2012).

The next section describes the validation of the developed ontology framework. This is done using a case study and expert judgement.

6.3 Ontology validation

Several ontology evaluations have been proposed, which can take either an implementation or a design viewpoint (Degbelo, 2017; Kumar and Baliyan, 2018). In the validation process, the semantic and syntactical correctness of the ontology is ensured, while it is also verified whether the ontology satisfies the targeted requirements. The scope of the present case study was exploratory– the aim was to present the development of a context-resolution service mechanism for industrial diagnostics, based on the design of a maintenance ontology focused on modelling the failure analysis of mechanical components. Therefore, it was considered appropriate to focus on a subset of evaluation criteria, namely usability through expert judgment, robustness, effectiveness, internal consistency and applicability, within the viewpoint of the targeted application case study.

6.3.1 Expert judgment

6.3.1.1 Usability

In addition to validating the framework through the use of a case study, expert judgment on the developed maintenance context ontology for the framework was sought. The use of expert opinion to validate an approach is a method commonly employed by researchers. Expert judgment and critique was sought to help not only validate the ontology, but also to provide insights for further refining the framework. In this respect, the usability of the framework was evaluated utilising the System Usability Scale (SUS) proposed by Brooke (1996), and subsequently employed in multiple studies (Hammar, 2017; Tan et al., 2017; Gregori, 2018). The SUS has proved to be highly effective in numerous applications, indicating that it can be used for a variety of different systems and forms of technology. It has been demonstrated that the results produced by the SUS are similar to those

of more substantial scales that further the understanding of user attitudes towards a given system's usability. Additionally, the SUS is capable of discriminating and identifying systems whose usability is high or inadequate.

Whilst the aim was to capture a broad array of expert views, the seven industry domain experts (DE1–DE7) successfully surveyed were sufficient to achieve the ROs. Some verbatim extracts are provided below. **Table 6-2** summarises the years of experience, backgrounds and qualifications of the experts.

Table 6-2: Information on the experts

NO.	Years of Experience	Area	Level of Education
DE 1	8	Mechanical engineer	PhD
DE 2	11	Maintenance engineer	Professional qualification
DE 3	13	Instruments & sensors	Professional qualification
DE 4	9	Maintenance engineer	Professional qualification
DE 5	10	Mechanical engineer	PhD
DE 6	15	Maintenance engineer	Professional qualification
DE 7	9	Maintenance engineer	Bachelor's degree

For the purpose of conducting the test, a basic questionnaire was developed for administration to the participants in order to determine the extent of their knowledge regarding ontologies, faults, the causes and effects of failure, and their maintenance terminology (see **Appendix B**).

A basic rating system was developed to enable an evaluation of the respondents' knowledge. Each response received a score ranging from 1 (no knowledge at all) to 5 (expert). Subsequently, three basic levels were determined for the different score ranges, based on the following ratings descriptions: scores under 15 were ranked as 'non-expert', scores 15–20 were ranked as 'good level of knowledge', and scores over 20 were ranked as 'expert'. **Table 6-3** shows the survey results.

Table 6-3: Level of knowledge scores

NO.	LOK 1	LOK 2	LOK 3	LOK 4	LOK 5	LOK 6	Level of Knowledge
DE 1	3	3	4	2	3	4	Good Level
DE 2	4	3	3	2	4	3	Good Level
DE 3	4	4	3	2	3	3	Good Level
DE 4	3	2	4	2	2	4	Good Level
DE 5	3	4	4	3	4	4	Expert
DE 6	4	5	5	4	4	5	Expert
DE 7	2	2	2	1	1	2	Non-expert

Previous test results had demonstrated that the majority of questionnaire respondents could be evaluated as having a good level of knowledge. Subsequent to gathering the survey results, the actual SUS test that was proposed by Casellas (2009) was conducted, after the model was presented via a brief presentation (see **Appendix C**).

The methodology used for the purpose of calculating the SUS scores was identical to that used by Tan et al. (2017) and Gregori (2018). To obtain the final value, the following adjustments were made to the scores: for questions with odd numbers, 1 was subtracted from the score (X-1), whereas for questions with even numbers, the score was subtracted from 5 (5-X). The scores from the questions with even and odd numbers were summed. Subsequently, this total was multiplied by 2.5, so that the final score could be presented in percentage form. Furthermore, Bangor et al. (2009) proposed a correlation between the SUS scores and the adjective rating scale for the purpose of assessing the SUS results, as shown in **Table 6-4**.

Table 6-4: Adjective rating scale – SUS score correlation

Adjective	SUS Score (up to :)
Worst Imaginable	12.5
Awful	20.3
Poor	35.7
Ok	50.9
Good	71.4
Excellent	85.5
Best Imaginable	90.9

Table 6-5 shows the evaluation results, including the 10 questions necessary for a SUS evaluation.

Table 6-5: Ontology usability evaluation

NO.	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	SuS score
DE 1	4	2	4	2	5	2	5	2	4	3	72.5 %
DE 2	5	3	4	2	5	2	5	2	4	2	75 %
DE 3	3	2	5	2	4	2	4	2	4	2	70 %
DE 4	3	3	4	2	3	2	4	2	4	2	62.5 %
DE 5	5	2	5	2	5	2	5	2	5	2	**80** %
DE 6	4	2	5	3	4	2	5	2	5	2	77.5 %
DE 7	3	2	3	3	2	2	3	2	3	2	55 %

The results showed that the individual scores were significantly over the acceptable threshold of 50.9%, with the average being 70.35% (good) and the peak being 80%. Hence, the usability of the model was successfully validated.

Nevertheless, it is important to note that, in the context of this validation, the primary focus was on qualitative not quantitative analysis. In this regard, the developed maintenance context ontology for the framework was peer-reviewed and published in the Journal of Frontiers in Computer Science (Mobile and Ubiquitous Computing section), under the title ‘Ontology-based Context Modelling in Physical Asset Integrity Management’. Publication is one of the most reliable validation strategies because the process involves feedback and

criticism. The work was reviewed and critiqued by referees from another leading journal.

6.3.2 Ontology applicability, robustness, internal consistency, and effectiveness

For the purposes of this study, a laboratory-based test rig was employed. When conducting maintenance inspection routines, queries are made on the designed maintenance ontology via a communication network utilising data obtained from the functioning machine, which can be achieved using the SPARQL Protocol and RDF Query Language, both being the most frequently used types of W3C.

To assess the model's robustness, a number of queries were constructed in SPARQL and tested on the ontology model. The process was considered satisfactory when all tests were shown to produce satisfactory responses in the given operation scenario. Moreover, the degree of detail is correlated with its fidelity, and is assessed by rigidly adhering to a consolidated process for ontology engineering, as well as an FMEA technique. In this study, the Ontology Development 101 process was used to guide the modelling.

To assess the reasoning consistency, the Pellet reasoner was employed. Real data collection from the shop floor, or simulated data with 'has current value' data properties, can be used to infer a component's health status and trigger alerts for decision-making, such as the prognosis of a failure and the scheduling of condition-based maintenance actions. In this regard, the SWRL language is used in object properties to construct transitive rules (Nuñez and Borsato, 2018), and new connections are applied to the classes that allow assertion inferences to be improved.

In this way, the ontological approach becomes scalable. Specifically, SWRL built-ins (SWRLb) allow further extensions within a taxonomy. This greatly enhances the model by allowing multiple arguments, according to specific real-world requirements, that enable greater expressiveness in OWL2 languages. A transitive property is considered in cases such as the following: if subclass component type (C1) has object property has mode, and subclass failure mode

(FM) has object property has cause (CA) related to subclass potential cause (PC), then subclass component type (C1) has the object property has cause (CA) related to subclass potential cause (PC). Then the SWRL rule is: has mode (?C1, ?FM1) failure mode (?FM1) component type (C1) potential cause (?PC1) has cause (?FM1, ?PC1) –> has cause (?C1, ?PC1). **Table 6-6** shows all the variables linked to each class of the maintenance ontology, for creating SWRL rules.

Table 6-6: Object properties using SWRL rules based on (Nuñez and Borsato, 2018)

Property	SWRL Rules
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Good"^^string) -> **has Velocity(?Gp1, "greater than or equal 0 and less than or equal 2.3**"^^string)
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Satisfactory"^^string) -> **has Velocity (?Gp1, "greater than 2.3 and less than or equal 4.5**"^^string)
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Alert"^^string) -> **has Velocity (?Gp1, "greater than 4.5 and less than or equal 7.1**"^^string)
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Alarm"^^string) -> **has Velocity (?Gp1,** "**greater than 7.1**"^^string)
Has Health	Measurement Location (?M1), greater Than Or Equal (?A, 0), has Current Value (?M1, ?A), less Than Or Equal (?A, 2.3) -> **has Health(?M1, "Good"**^^string)
Has Health	Measurement Location (?M1), has Current Value (?M1, ?B), greater Than (?B, 2.3), less Than Or Equal (?B, 4.5) -> **has Health (?M1, "Satisfactory**"^^string)
Has Health	Measurement Location(?M1), greater Than (?C, 4.5), has Current Value(?M1, ?C), less Than Or Equal (?C, 7.1) -> **has Health (?M1, "Alert**"^^string)
Has Health	Measurement Location (?M1), greater Than (?D, 7.1), has Current Value (?M1, ?D) -> **has Health (?M1, "Alarm**"^^string
Has Warning	Has Health (?M1, ?A), equal (?A, "Good"^^string), Measurement Location (?M1) -> **has Warning (?M1, "Collect new data in 3 months**"^^string)
Has Warning	Has Health (?M1, ?B), Measurement Location (?M1), equal(?B, "Satisfactory"^^string) -> **has Warning(?M1,** "**Collect new data in 1 months**"^^string)
Has Warning	Has Health (?M1, ?C), Measurement Location (?M1), equal (?C, "Alert"^^string) -> **has Warning (?M1, "Schedule Condition-based Maintenance**"^^string)
Has Warning	Has Health (?M1, ?D), Measurement Location (?M1), equal(?D, "Alarm"^^string) -> **has Warning (?M1, "Turn off equipment**"^^string)
Is Caused By	Has Health (?M1, ?A), equal (?A, "Good"^^string), Measurement Location(?M1) -> **is Caused By (?M1, " Everything is ok** "^^string)
Is Caused By	Has Health (?M1, ?B), Measurement Location (?M1), equal (?B, "Satisfactory"^^string) -> **is Caused By (?M1, " Loss of lubricant - Housing bore out of round - Corrosive agents - Distorted bearing seals**"^^string)
Is Caused By	Has Health (?M1, ?C), Measurement Location (?M1), equal (?C, "Alert"^^string) -> **is Caused By (?M1, " Excessive loading (cyclic), misalignment** "^^string)

Property	SWRL Rules
Is Caused By	Has Health (?M1, ?D), Measurement Location (?M1), equal (?D, "Alarm"^^string) -> **is Caused By (?M1, " Misalignment - Housing bore out of round - Unbalanced/excessive load**"^^string)

As part of the SWRL rules in the suggested ontology, ISO10816-3:2009 was utilised for the evaluation of the data gathered from the vibration measurements and analysis. The test rig used in the validation study was regarded as a mid-level asset in an asset hierarchy that included a rolling-element bearing (at a lower level in the asset hierarchy), and that also included an accelerometer acting as a transducer in the data collection process. The resulting assessed parameters could include the velocity of the vibration in mm/s root mean square (RMS), with the measurement sites defined by standard MIMOSA VB-00, while the operating zone limits were based on the ISO10816–3:2006 standard.

Assume that, when the data for the rolling bearing part give a mm/s RMS value of between 0 and less than or equal to 2.3, a ‘good’ notification is displayed. When values exceeding 2.3 but below 4.5 are detected, a ‘satisfactory’ notification is shown. A value between 4.5 and 7.1 would trigger an ‘alert’ notification, and values in excess of 7.1 would cause an ‘alarm’ notification, which would instantly terminate the machine’s operation.

These values change from one machine to another. The ISO10816-3 standard explains that these are only relevant for specific types of machines. So, the value 2.3 is only applicable to machines in Group 1 (large machines with rated power over 300 kW but below 50 MW, or electrical machines with a height of more than 3.15 m). The same standard indicates that, for smaller machines, the value is 1.4 instead of 2.3. Given this, let us assume that a value of 4.7 mm/s RMS is recorded. This is fed through the ontology, activating the Pellet plugin reasoner in the Protégé ontology editor, producing an ‘alert’ outcome, prompting a recommendation to ‘schedule condition-based maintenance’, as illustrated in **Figures 6-8 and 6-9**. Once an ‘alert’ warning has been issued, it is then important to associate it with the diagnostic information for the analysed mechanical component, linking the identified failure mode to potential causes (**Figure 6-10**). In this way, the maintenance intervention becomes context-depended, and

consequently more focused and relevant to the identified context of the monitoring situation.

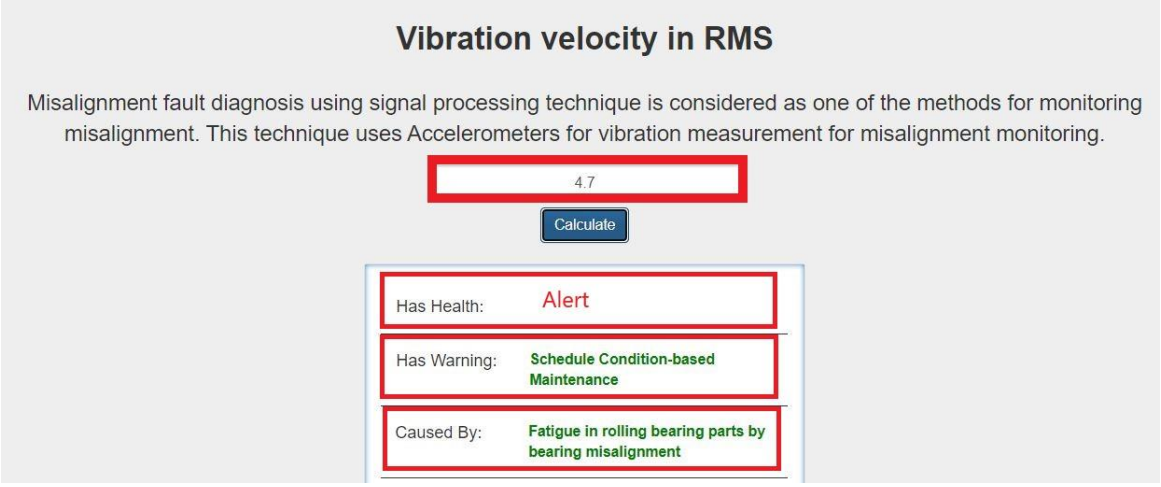


Figure 6-8: Vibration velocity in RMS on context-based adaptation interface

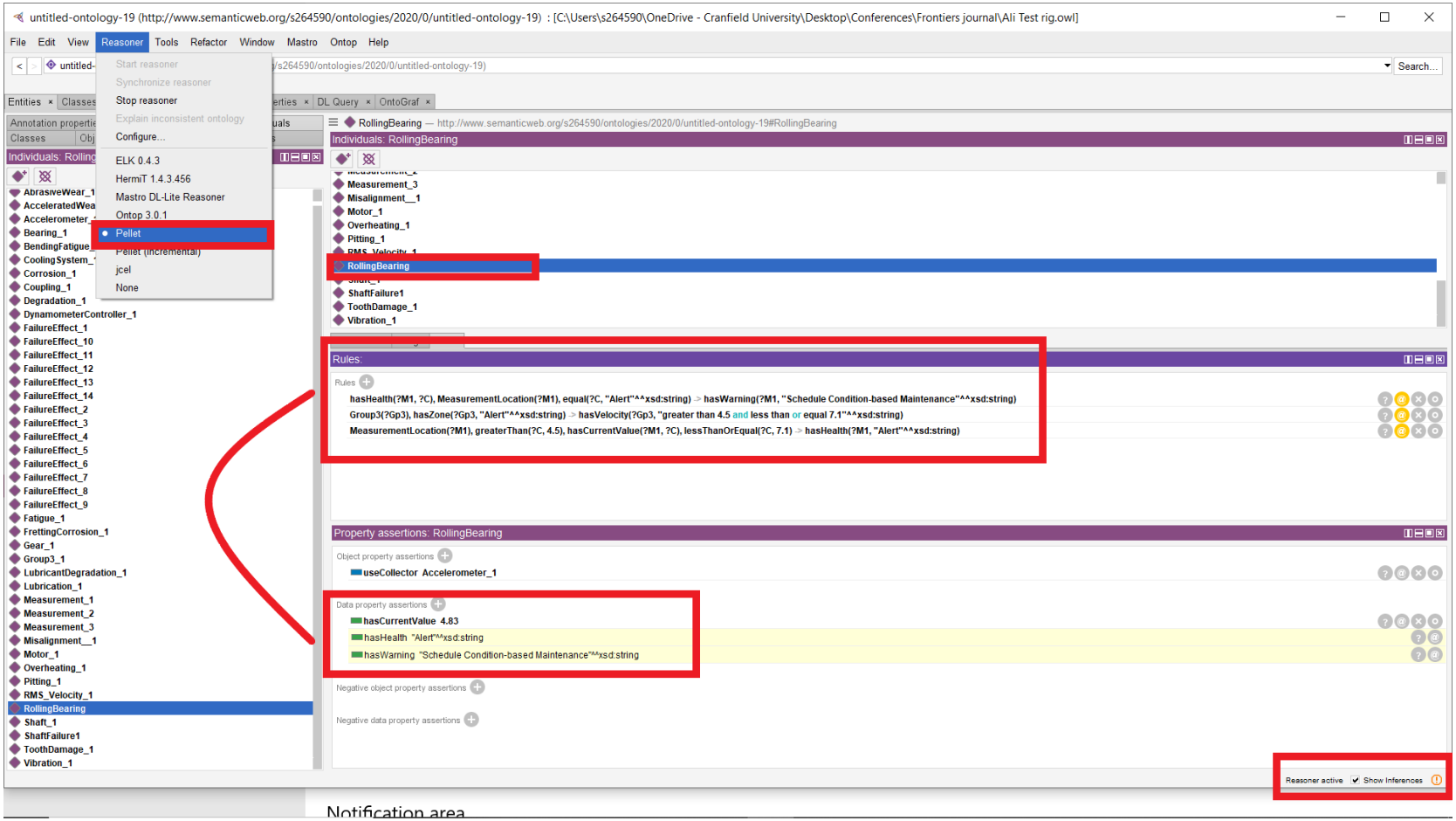


Figure 6-9: SWRL rules for generating a warning

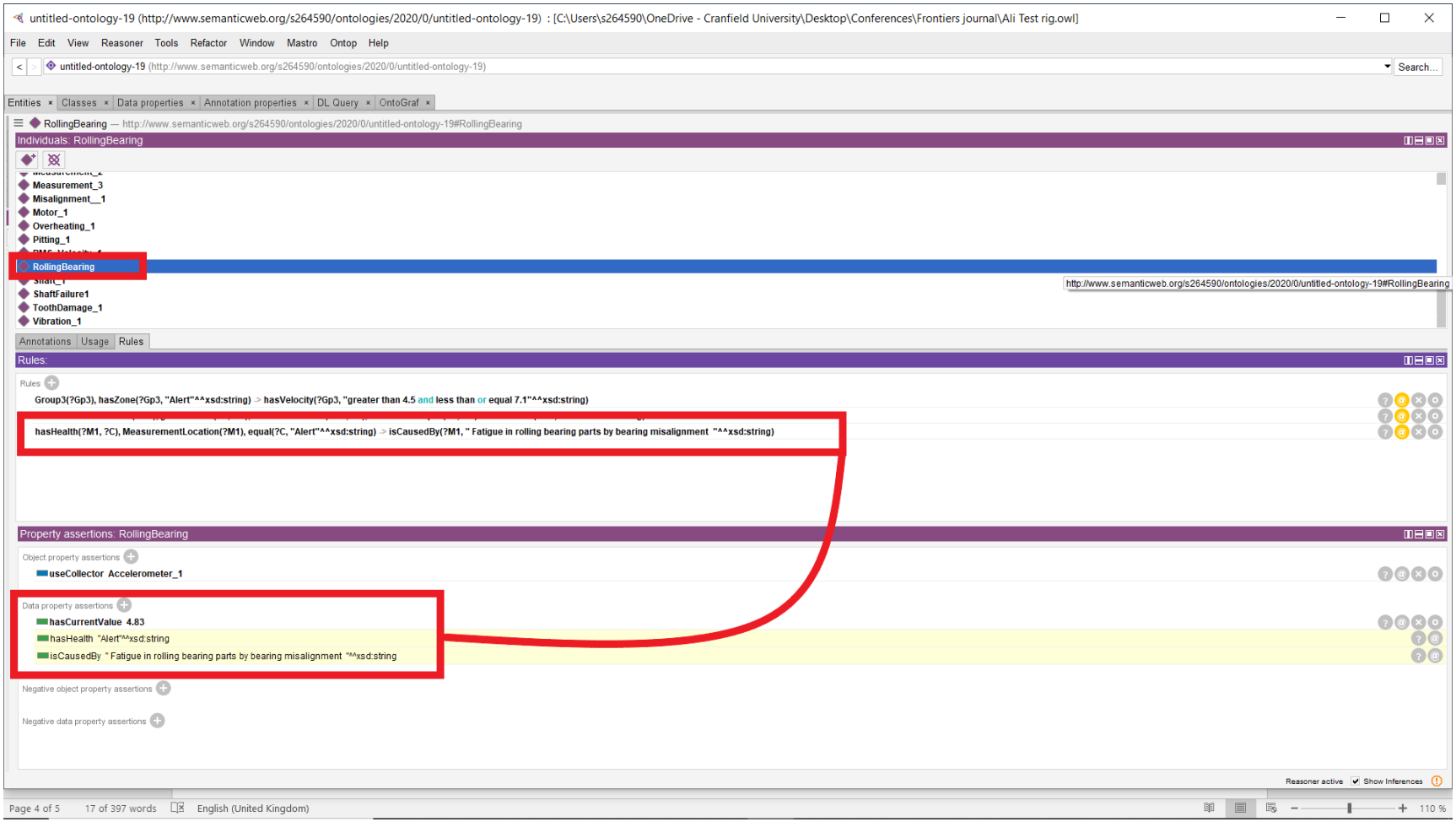


Figure 6-10: SWRL rules for generating potential cause

The Pellet reasoner was applied in this work because it can detect inconsistencies and verify class hierarchies, range, domain and conflicting disjoint assertions. The Protegé editor provides a warning with a red triangular alert symbol when a consistency error occurs. The Pellet reasoner is subsequently activated, as shown in **Figures 6-9 and 6-10** in order to detect any inconsistencies. It was employed here to make sure that no inconsistencies were present.

In summary, while simple, single-parameter threshold-based rules might be easy to interpret, they do not often hold in practice. Instead, more complex, multi-parameter rules are more likely to apply. The reasoning process can replace simple rules with the activation of more complex decision functions, and these can be produced as a result of machine learning over monitoring historical data. The value of the described process is that it sits at a higher level of abstraction, and can therefore work with different lower-level computational rules.

The following section is aimed at validating the developed architecture. This was done through the simulation of two different scenarios.

6.4 Architecture Validation

Within the IoT system, there is a connection between the sensors, actuators and smart devices and the Internet. Numerous technologies play a role in this process, particularly by making the connected devices work together. Cloud computing has specific relevance, as it enables hosted services to be delivered. Certain applications necessitate that the interaction among IoT devices and the cloud for remote monitoring services should be analysed in detail. In particular, research in the field of monitoring services is confronted with issues that include how such a large group of devices should be governed, as well as how meaningful data can be provided in real time that depicts the condition of a machine over time. The next section of this chapter will focus on the design of a simulation, through two different scenarios, to illustrate the operation and utility of the proposed architecture for the integration of the IoT and cloud computing for industrial monitoring services.

6.4.1 Scenario 1 “Building Management System (BMS)” (Early Prototype)

The following section describes a Building Management System (BMS) for automating Smart Buildings based on the developed architecture presented in Chapter 4. The primary objective of a BMS is to enhance the comfort of those in the building by ensuring that it is consistently maintained in the intended condition, as well as to decrease the use of energy by preventing scenarios involving the excessive use of such; for example, by forecasting the time at which an individual will enter a room, the temperature can be adjusted in advance, or by identifying when a room will be unoccupied, lights can be switched off that have been left on unintentionally. In this way, BMS scenarios can be used to create variations in the context, including rules of thumb for determining context-specific responses (control actions).

In this scenario, Cranfield Thingsboard platform was utilised to validate the applicability of the proposed architecture. ThingsBoard is an open-source IoT platform that allows IoT projects to be rapidly developed, managed and scaled. It enables users to develop rich IoT dashboards for the purpose of visualising data and then applying controls on remote devices in real time. ThingsBoard facilitates the connectivity between devices through developed application layer protocols (e.g. MQTT, CoAP and HTTP), and provides for their deployment via the cloud.

This section presents an explanation regarding the data generation process, as well as the specific elements in the building that are being monitored. Due to the fact that deploying multiple sensors in a building can be expensive, they were simulated for the purpose of scaling the system and creating a more realistic scenario. This is followed by a description of the transformation of the data into a conventional format, and its subsequent uploading onto the cloud platform. Finally, the usage of the data via the building management application is explained. **Figure 6-11** shows the elements that make up the system, divided into different layers.

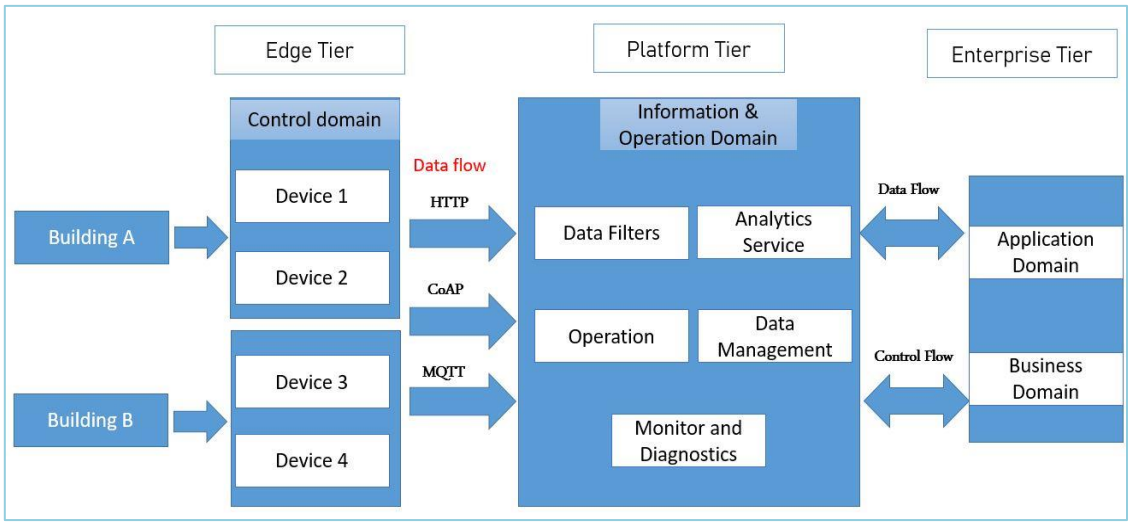


Figure 6-11: Implementation of high-level architecture

To date, BMS has generally only been used in the industrial and commercial sectors, although recently it is becoming more frequently implemented in large domestic cases. In the commercial and industrial HVAC sector, BMS has been widely employed for some years. The types of BMS used in the commercial sector include simple control units to total building control in multi-floor, multi-functional buildings, as shown in **Figure 6-12**.

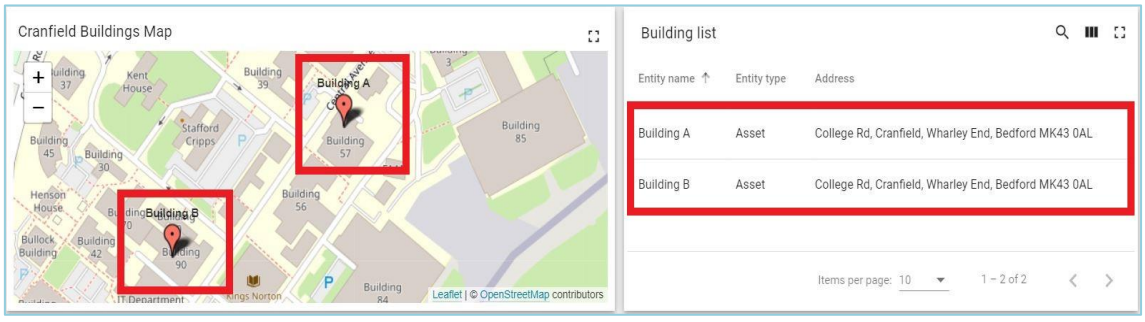


Figure 6-12: BMS system for monitoring a group of buildings

Figure 6-12 shows that a BMS system can be used to monitor a group of buildings in different locations. In this scenario, the buildings are **Building A** and **Building B**, with the BMS being used to control and monitor a variety of services inside a building, such as heating and ventilation systems, as well as water consumption, as shown in **Figure 6-13**. Additional functions, including monitoring the level of occupancy in the building, can be employed for controlling features

such as lighting and heating demand, so that the activation of such functions only occurs in the event that parts of the property are occupied or being used.

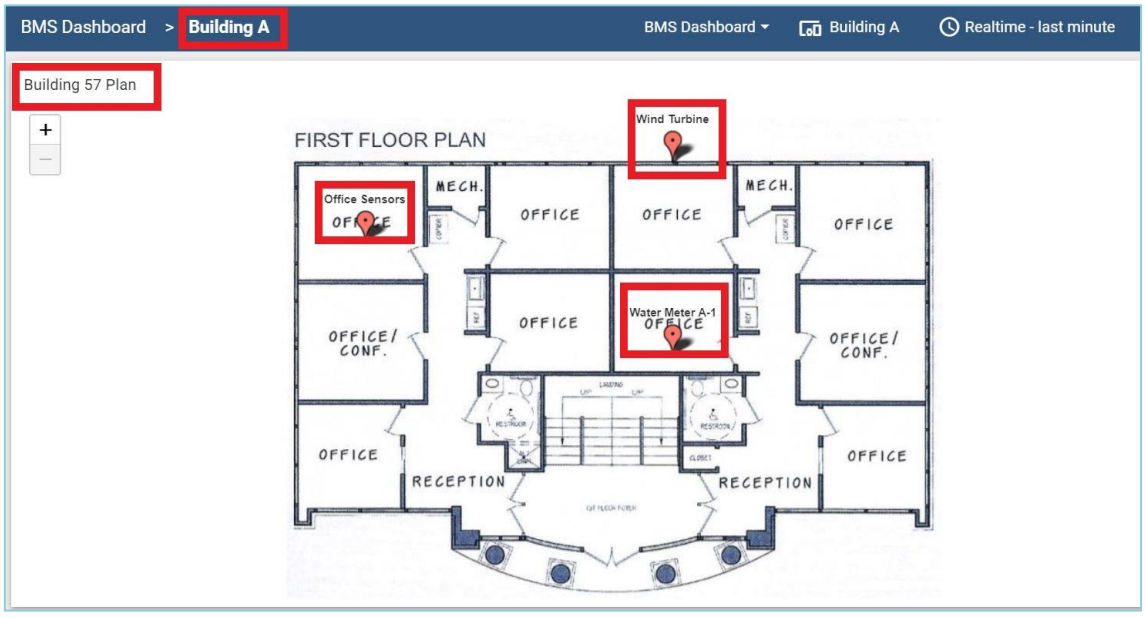


Figure 6-13: BMS for building A

The initial step in enhancing a property with intelligent features is to monitor all the required components it contains. For buildings, elements that have importance include lights, HVAC systems, CO_2, light intensity, water, energy and room occupancy. It is also necessary to monitor the environment to determine whether it can be exploited; for example, turning lights off if the external luminosity is sufficiently high for internal use.

Certain components are equipped with small sensors that have the capability of acquiring the data required to determine the state. Because deploying multiple sensors in a building can be expensive, they were simulated for the purpose of scaling the system and creating a more realistic scenario. With regard to environmental data, humidity, temperature and luminosity, sensors allowed the respective values to be read. To mimic the remaining components of the system, data was produced using software that can create identical packets to those of physical elements.

After reading the sensor data and then encapsulating it in a packet with the shape, it was transmitted to the ThingsBoard platform. Messages are received by

ThingsBoard gateways through various protocols, such as HTTP, MQTT and CoAP. As gateways are boosted by an Internet connection, raw messages are forwarded to a central server tasked with aggregating and standardising data.

When combined with wireless monitoring and control systems, BMS will be able to significantly enhance the systems currently being used. However, considerable amounts of data are generated by such systems, and these need to be monitored, logged and analysed. In industrial and commercial environments, these collected data are given to the facilities manager, who conducts an analysis on the data and then makes operational decisions accordingly. There is an expanding market for software instruments that are able to analyse and present monitored data in easily-understandable and rapidly-available formats. Technologies, such as web dashboards, are meeting this gap in the market.

Web dashboards are capable of monitoring data from various different sources, and subsequently displaying the data on-screen, similarly to the manner in which information is presented to car drivers on their dashboards. Graphical and coloured interfaces are generally used to present such data, which allows a facility's manager to rapidly determine and deal with problems or irregularities that might emerge in the functioning of the system they are controlling, as shown in **Figure 6-14**.

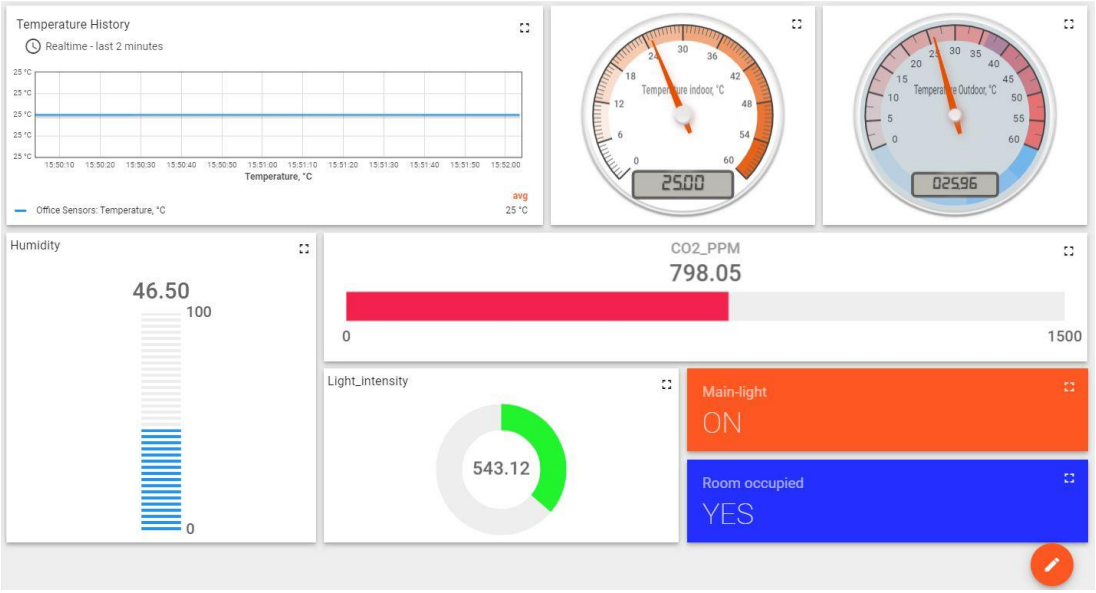


Figure 6-14: Building A dashboard

Collected data are presented by such dashboards via aesthetically-satisfying interfaces that are user friendly and easily understandable. Although it is possible for ‘abnormal’ readings to occur when data telemetry is processed and collected, such readings are not within the expected range of telemetry readings for the device. Additionally, it is possible to create alarms via ThingsBoard, which are activated in the event that the telemetry sent by an IoT device falls outside the specified range, as illustrated in **Figure 6-15**.

Figure 6-15: Alarms system

In general, the proposed application consists of a pair of differentiated components. First, the BMS is tasked with directly receiving sensor data through the subscriptions performed to the various sensors within the property. Once a new message is stored in the middleware database, it is also forwarded to the application, permitting the BMS to act correspondingly, where appropriate. Nevertheless, as previously noted, due to the fact that deploying and testing this type of scenario can be expensive, simulation was selected as the second component in order to acquire results that were realistic regarding the benefits of the property improved with intelligent features.

The entire system behaves in the following way. The BMS is fed with both actual and software-generated sensor data by the BMS. Subsequent to the data reaching the application, their state is modified by simulated components so that it can be synchronised with the matching sensor. For example, the room light turns on automatically when motion is detected by a motion sensor (motion sensor light), as shown in **Figure 6-16**.

Figure 6-16: Occupancy based on motion detection

Figure 6-16 illustrates a light bulb with a motion sensor switch. A light is installed in the room, and includes a switch and a motion sensor. When motion is detected, the switch closes the circuit and the light turns on. Based on this pattern, the synchrony between the actual and virtual sensors and the simulated building elements is maintained by the simulator. Resultantly, if it is detected by the simulator that an actuation is required, it automatically alters the element's state, and all of the necessary software-defined sensors are updated so that the synchronism can be maintained.

The final feature of the simulator relates to the intelligent functionality of the BMS. In other words, the system must be capable of detecting whether an action that enhances comfort, while potentially increasing energy production, can be continued by examining the property's state at every moment. Recent advancements in building-integrated wind-turbine technologies have made them more reliable, enhanced their efficiency at reduced wind speeds, and lowered their capital expenses. Micro wind turbines can be applied at the building scale–termed 'building-integrated wind turbines'–for the local production of electricity. In this scenario, the wind turbine is represented as an asset, as indicated in **Figure 6-13**. The wind turbine has two devices installed—a wind-direction sensor and a rotating system. In this scenario, the rotating system is responsible for changing the direction of the wind turbine based on the wind direction, as displayed in **Figure 6-17**.

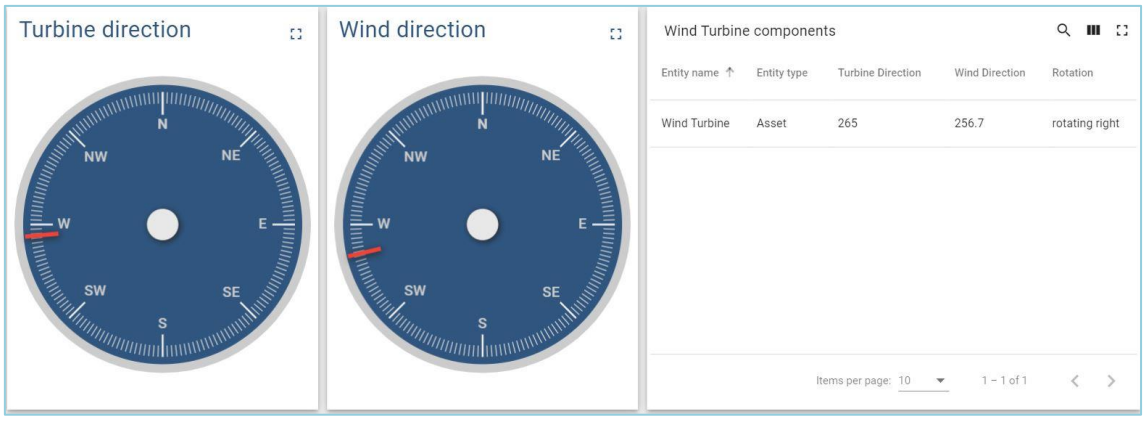


Figure 6-17: Wind turbine rotated based on wind direction

In the current scenario, the representation of context information is achieved via values allocated to properties that characterise all entities that have relevance to an application. The cloud platform allows the defining of assets, devices, rule chains and dashboards, among others. The physical 'things' that the IoT solution is targeting are defined as asset'. Device 'things' are defined as devices. 'Rule chains' are essentially the means for designing IoT data process workflows. Data can be forwarded to external systems, or actions can be triggered using custom logic via the use of rule chains. The sharing of context information facilitates the process of producing, gathering, publishing and consuming context information by users at large scales, and it can be exploited for the purpose of transforming all applications into 'smart' and 'aware' forms. The section below moves on to consider the physical implementation of the proposed scenario.

6.4.1.1 Physical Implementation

This physical implementation involved the detection of real-time temperature, relative humidity and object distance of a building at predetermined intervals, using an embedded computing device (Raspberry Pi). In this case, the Raspberry Pi microcontroller is the monitoring node, while the sensor used is a DHT22 temperature/humidity sensor, as illustrated in **Figure 6-18**. The connection between the sensor and the Raspberry Pi device was achieved using a jumper wire. The Python language was used for programming the Raspberry Pi kit. The ThingsBoard platform facilitated the real-time monitoring of the sensor data. The

data processed by the Raspberry Pi was continuously updated on the cloud server, with the users being able to access the stored data.

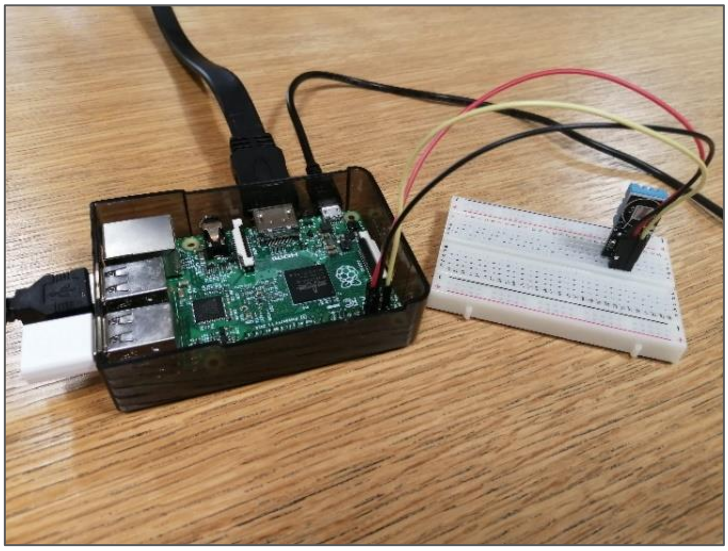


Figure 6-18: DHT22 sensor connected to a Raspberry pi 2 (microcontroller)

For this purpose, the code for the DHT22 sensor was designed in Python (see **Appendix D**), although different languages, such as C, C++ and Java, could also have been utilised. Python was employed for this application because it offered certain benefits. The length of a Python programme is generally 3–5 times less than a corresponding Java programme. This distinction is due to Python's integrated, high-level data types, as well as its dynamic timing. It is intended to provide high readability. Python is a basic, dynamic, interpreted and object-oriented programming language, and Python interpreters allow Python code to run on multiple different systems.

Data from Raspberry Pi was sent to a ThingsBoard server, and was updated and stored after each second (the processing time of the sensor is 1 second, and so the data was in fact updated every second). The update time can be increased, but not reduced. Data from the sensor was visualised using different widgets, as shown in **Figure 6-19**.

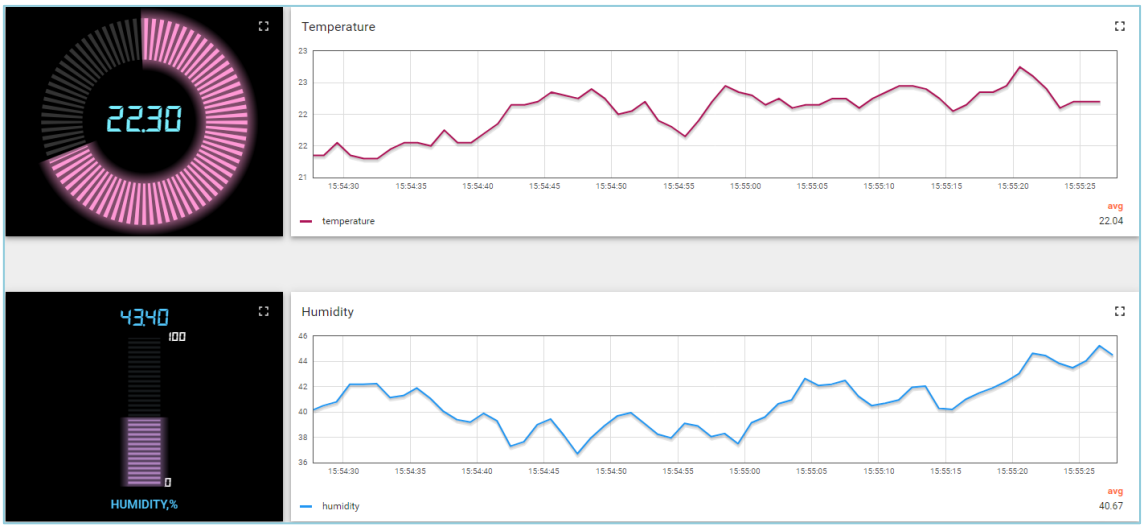


Figure 6-19: Temperature and Humidity readings

The development of small, inexpensive sensors has paved the way for the emergence of systems that can monitor for various fields of application in real time. The IoT is an innovative field of IT that facilitates connections between a number of these heterogeneous devices via distinct network protocols to enable large-scale interoperability. It was shown by the physical implementation how it is possible to integrate certain open-source software tools for the purpose of collecting, monitoring and processes data streams delivered in real time by sensor devices. The proposed architecture can be used in a variety of monitoring scenarios for studying the patterns of certain key parameters, as well as to trigger alarms when problematic situations occur. The following section moves on to consider a mechanical gear rig that was used in the simulation.

6.4.2 Scenario 2 “Machine Monitoring System”

To test the applicability of the proposed architecture, a physical gearbox test rig was employed (**Figure 6-20**). Digital twinning of the gearbox was implemented in a local cloud-based deployment of an IoT platform (ThingsBoard). The rig was instrumented with industry-grade sensing, data acquisition and networking, with both edge and cloud-based computing support for a complete data-process workflow.

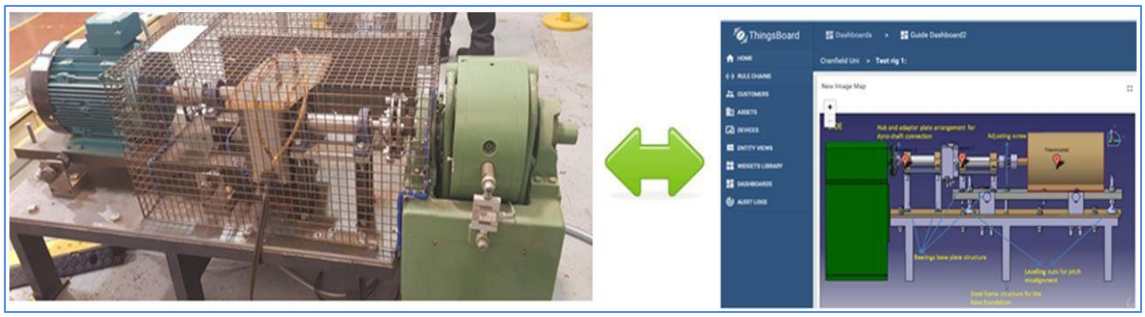


Figure 6-20: Physical gearbox transmission test rig for emulating misalignment cases

In this case study, the most common effect of gearbox misalignment was vibration, and loss of power transfer through the gearbox. The vibration affected the entire machine, with the most prominent points being the bearings, the loss of transfer easily being captured by the difference in RPM between the driving motor and the loading dynamometer. The reason for vibration being the main sign of an issue in such a machine is because the machine consists of several high-speed rotating components.

The wearing of gear teeth is generally caused by gear misalignment, excessive loading and a lack of lubrication. Degradation of the bearings is caused by wearing of the gear teeth and the impact of gear vibration being transferred to the shaft, and then to the bearings. If the shaft of the gear is short and hard, and the bearings are situated in close proximity to the centre, where meshing of the gears occurs, this can be a source of vibration, which can be measured by placing sensors at the bearings, as shown in **Figure 6-21**.

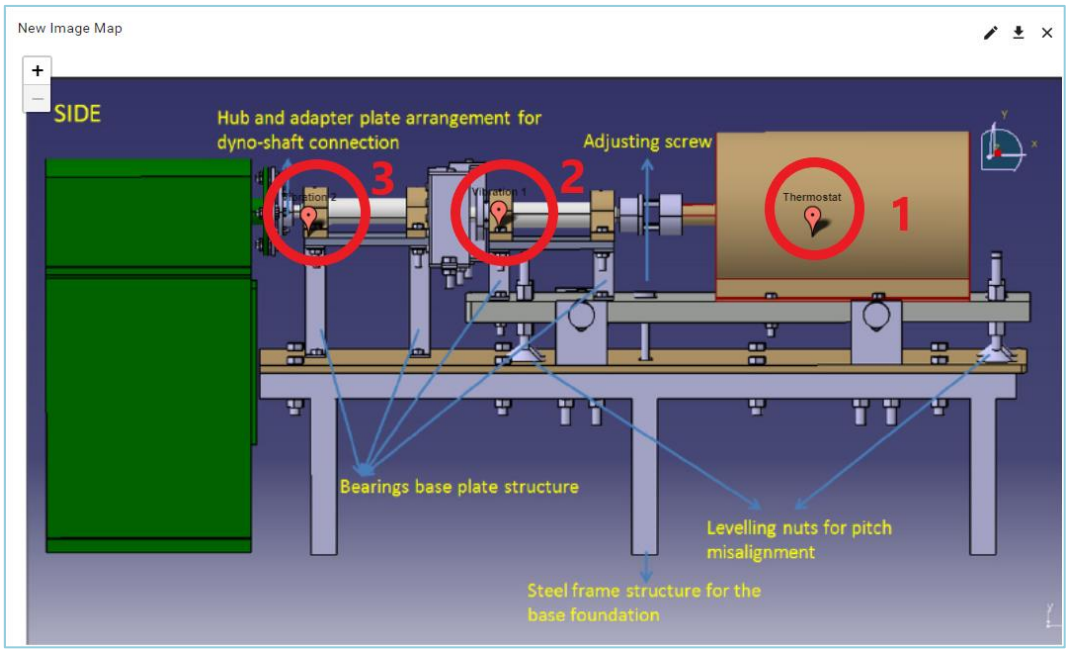


Figure 6-21: Draft layout of the gear test rig presented on the ThingsBoard platform

For this scenario, the gear test rig was defined as an ‘asset’ and the sensors as ‘devices’. In summary, the scenario was aimed at designing and developing an IoT application that could collect temperature and vibration readings from multiple sensors deployed in a gear test rig. For simplification, two different types of sensors were deployed, as shown in **Figure 6-21**. These sensors included a single thermometer for generating temperature data, and two accelerometers for measuring the angular velocity (rate of change of the angular position over time, along the X and Y axes). Thus, three devices were created—‘thermostat’, ‘vibration1’ and ‘vibration2’. Once the ‘thermostat’ icon installed on the gear test rig was clicked, the system allowed the end-users to monitor real-time temperature data, as shown in **Figure 6-22**.

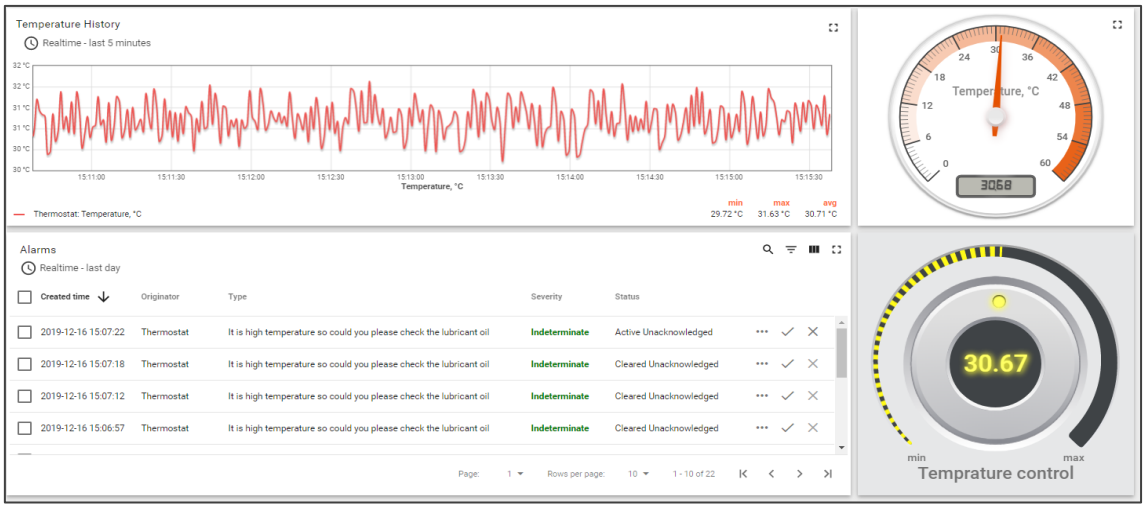


Figure 6-22: Real-time temperature data

Most critical problems result in the overheating of the motor or dynamometer, or in vibration propagating throughout the machine due to gear misalignment. This is why temperature and vibration sensors were deemed the most appropriate to capture those particular potential failures. **Figure 6-22** (top half) shows the graphical user interface for the real-time temperature readings in different widgets (chart and analogue gauges). Charts are beneficial for visualising real-time or historical data with a time window. The bottom half of **Figure 6-22** presents the temperature control and alarms. Temperature control can be used to effectively visualise the present state, as well as to send RPM commands to target devices. For this scenario, the temperature control was used to increase and decrease the temperature readings.

On the other hand, once the ‘vibration1’ icon installed on the gear test rig (**Figure 6-21**) was clicked, the system allowed end-users to monitor a time-series vibration signal, as indicated in **Figure 6-23**. Vibration data is one of the most commonly used techniques for detecting abnormalities in a machine. To measure vibration, three different types of sensors can be used–motion, speed and accelerometer. An accelerometer was chosen for this study because it was the solution with the greatest dynamic range and the largest frequency range. Other solutions are not suitable for industrial use.

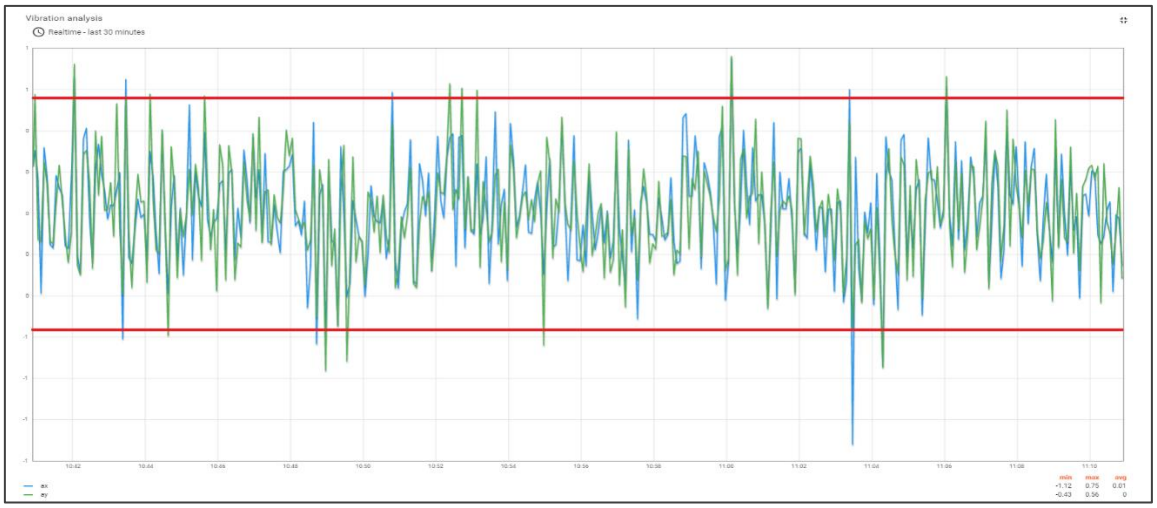


Figure 6-23: Time series of a vibration signal

Comma-separated values (CSVs) were used here as a method of collecting data because of the flexibility offered. CSVs are broadly used for exchanging data-table data among applications with distinct architectures for the purpose of solving issues involving the interoperability of data formats that are not compatible. In general, the format is defined on the basis of the predetermined standards of both parties, and a clear format standard has not been established. For this study, the data used to feed the platform was from previous tests by Prof Muhammad Khan (Cranfield University), including data from displacement sensors and accelerometers. In order to visualise the collected dataset, node.js code was developed (see **Appendix E**).

After receiving the data, the platform stored the data in the memory and displayed the vibration curve through the chart, as shown in **Figure 6-23**. Depending of the values of the data with regard to the ranges of values (descriptors), the interpretation rules defined the condition of the asset in terms of the associated failure mode. Based on that, the system provided a reminder and an issue or a warning. This could be used as a simplified interpretation rule to monitor misalignments depending on the load.

In summary, this section described the simulation environment which has been used to test the proposed architecture. The proposed architecture is aimed to introduce context awareness to enhance remote monitoring services in

manufacturing environments. Functional test requirements were identified that need to be passed by the key components of the proposed architecture that will serve quality-assurance purposes. These functional tests (unit, code, integration, communication and system testing) were designed to investigate the functionality and applicability of the key components of the proposed architecture. In light of that, this section also focused on simulation design through two different scenarios (building management system and machine monitoring system) to illustrate the operation and utility of the proposed architecture for the integration of the IoT and cloud computing for industrial monitoring services. The context information management is used by the proposed architecture to help the user to get a particular service without much user intervention. The application of the architecture on simulation design through two different scenarios respectively provide some assurance regarding the validity of the developed architecture. Moreover, the proposed scenarios showed the performance of the proposed architecture by simulating the system to fetching the services automatically for the user.

6.5 Chapter Summary

The validation process incorporated in the development of the framework and architecture for introducing context awareness to enhance remote monitoring services in manufacturing environments was outlined in this chapter. This represented phase four of the multi-phase research methodology presented in Section 3.7 and RO 4. **Table 6-7** summarises how the elements of the proposed framework (Figure 4-14) are validated.

Table 6-7: summarises how the elements of the proposed framework are validated.

Framework layer	What is validated?	How is validated?	Where is validated?
Edge layer	Functional test requirements were identified that need to be passed by the key components of the proposed framework that will serve quality-assurance purposes. These functional tests (unit, code, integration, communication and system testing) were designed to investigate the functionality and applicability of the key components of the proposed framework.	The Raspberry Pi microcontroller is the monitoring node, while the sensor used is a DHT22 temperature/humidity sensor. The Python language was used for programming the Raspberry Pi kit. The ThingsBoard platform facilitated the real-time monitoring of the sensor data.	6.4.1.1 "Physical implementation"
Context information management layer	A maintenance context ontology for the framework has validated focused on modelling failure analysis of mechanical components, so as to drive monitoring services adaptation.	The ontology development was applied to a physical mechanical-transmission test rig. The focus of the validation was specifically applied to five ontology quality features– robustness, level of detail, effectiveness, internal consistency and applicability. In addition to the case study validation process, expert judgement was sought, which was generally positive.	6.3.1 "Expert judgment". 6.3.2 "Ontology applicability, robustness, internal consistency, and effectiveness"
Application layer	The applicability of the key components of the proposed framework.	For the virtual prototype, different scenarios have been created. These scenarios focused on the design of simulation through two different scenarios ("Building Management System" and “Machine Monitoring System”) to illustrate the operation and utility of the proposed framework for the integration of IoT and cloud computing for industrial monitoring services. The proposed scenarios showed the performance of the proposed framework by simulating the system to fetching the services automatically for the user.	6.4.1 Scenario 1 “Building Management System (BMS)” 6.4.2 Scenario 2 “Machine Monitoring System”

The ontology development was applied to a physical mechanical-transmission test rig. Queries were raised in terms of the resolution of the monitoring service context to determine the failure mode of the test rig and its potential causes, in addition to the relevant measurement parameters. The outcomes can be used in other industrially-relevant application scenarios to drive maintenance service adaptation. While the application focus was quite specific, the ontology abstraction level was actually such that it could also be implemented in other application cases, as it offers a sound baseline for further customisation or extensions.

The focus of the validation was specifically applied to five ontology quality features–robustness, level of detail, effectiveness, internal consistency and applicability. In addition to the case study validation process, expert judgement was sought, which was generally positive. The next chapter concludes the study, and offers research limitations and ideas for future work.

7 CONCLUSIONS AND FUTURE WORK

This chapter provides a summary of the research project, highlighting how the ROs and RQs have been achieved. The contribution to knowledge from undertaking this research project is then discussed, before areas for future work are suggested, based on the limitations of this study.

7.1 Contribution to the knowledge

With the stated aim and objectives of this research, this project was positioned to make several contributions to knowledge, with implications at three levels– academic, scientific and business.

- **At the academic level**, the systematic literature review would provide an update to previous reviews on the topic of context management in the IoT, creating a potential checkpoint for researchers intending to perform a study on this topic. In addition, and most importantly, the appropriate factors used in context acquisition, modelling, reasoning and dissemination were critically analysed so as to develop a framework and an architecture that could introduce context awareness in order to enhance monitoring services in manufacturing environments. According to the available literature, as well as the researcher knowledge base, this method of combining IoT and cloud computing with context awareness for industrial monitoring services had not previously been covered or investigated.

The core **academic contributions** of this research include;

- A context awareness framework and architecture, developed to enhance remote monitoring services in manufacturing environments. The framework and architecture apply context-aware computing to deliver solutions and address key challenges that IoT-enabled monitoring services need to handle, specifically dealing with ways in which the context can be modelled, processed and disseminated for remote monitoring services, how this impacts the service discovery solution, and what an appropriate taxonomy of ontology is.

- A context maintenance ontology for the framework, developed by focusing on the modelling failure analysis of mechanical components so as to drive monitoring services adaptation. The proposed ontology for the context resolution mechanism is relevant to the failure analysis of mechanical components, and the terminology and relationships between the concepts are structured on the basis of relevant standards so as to adapt relevant diagnostics or maintenance actions in a condition-based maintenance setting.
- **At the scientific level**, a novel classification of the relevant literature has been produced, following a survey and critical analysis. To that end, three conference papers have been published. The first was presented at the 13th World Congress on Engineering Asset Management in Norway in 2018. It offers a survey and analysis of the recent literature that addresses context management in IoT. The second was presented at the 7th International Conference on Through-life Engineering Services, and was aimed at presenting a context-awareness IoT framework for industrial monitoring services. The third was presented at the 4th IFAC AMEST 2020 Workshop on Advanced Maintenance Engineering, Services and Technologies. In this paper, ontology-based approaches for semantic maintenance were proposed as a data and service mediation mechanism. Ontology-Based Context Modelling in Physical Asset Integrity Management was published in a Frontiers journal. This presents the system framework and the ontology development process, based on established practice and maintenance vocabulary standards. Partial results from these publication have been used in the thesis chapters.
- **At the business level**, the outcomes of the work can be used in other industrially-relevant application scenarios to drive maintenance service adaptation. Context adaptive services can help manufacturing companies in better managing the value of their assets, while ensuring that they continue to function properly over their lifecycle. This allows managers and operators to assume control and make more appropriate decisions based on data, thus facilitating improved, real-time communication among

managers and the shop floor. Furthermore, the proposed framework will help those companies to interpret the situation of a monitoring service and its associated data acquisition in order to identify various faults at the monitoring level and to propose actions contextually relevant to the identified situation.

To achieve the research aim, the study had four ROs, which are addressed below.

7.1.1 Review of the Research Objectives

RO1 - Analysing Current Practices of Context Lifecycle Management

The first RO was aimed at analysing the current practices of context lifecycle management, and identifying the appropriate factors that are used in context acquisition, modelling, reasoning and dissemination for IoT-enabled industrial monitoring services. To achieve this RO, a systematic literature review was conducted. In light of that, the appropriate factors used in context lifecycle management were critically analysed, and published in a conference paper. Furthermore, an extensive literature review was undertaken which highlighted that IoT has expanded the range of applications with substantial needs for context management, and this is reflected in the focus of the relevant studies. Nonetheless, while substantial research efforts have been devoted to context lifecycle management in web-based, mobile and ubiquitous computing, including IoT-enabled computing, little attention has been given to translating these advances into tangible progress in industrial monitoring services.

A gap identified from the literature was that most applicable context-modelling and reasoning techniques that had been examined were ontology-based. However, the studied approaches lacked some expressiveness concerning the representation of knowledge for monitoring services in manufacturing environments. Because contemporary computational infrastructures facilitated by the IoT are dynamic in nature, it is necessary for applications to be aware of the contextual data of interest, with an ability to function with minimal human

intervention. Hence, context awareness is now considered to be a critical factor in the provision of adaptive services in IoT settings. Thus, the aim of this research project was to contribute by developing a context-aware framework to enhance monitoring services in industrial environments as a means of addressing challenges related to information complexity, as well as to integrate data with domain knowledge in industrial monitoring applications.

RO2 - Developing a Context-aware Framework to Enhance Monitoring Services

Building upon the understanding obtained from the extensive literature review, the second RO was meant to design and develop a framework and an architecture that would introduce context awareness to enhance remote monitoring services. In order to accomplish this RO, the multi-phase research methodology adopted helped to facilitate the development of a context-aware framework to enhance monitoring services in industrial environments. In this respect, the knowledge and understanding gained from the extensive literature review and the experiments (early prototype and virtual prototype) was instrumental in the development of the framework.

The developed architecture has four distinct but interconnected viewpoints. This consisted of four viewpoints–business, usage, functional and implementation. The business viewpoint determines the manner in which the IoT system accomplishes the established goals via its mapping to basic system functionalities. The usage viewpoint deals with the concerns of anticipated system utilisation, and is usually denoted by series of activities incorporating human or logical users, which provide its predetermined functionality in eventually accomplishing its basic system capacities. The functional viewpoint concentrates on the functional elements within a system, their interconnectedness and formation, the interfaces and interrelations, and the associations and interconnections of the system with components in the external environment. Finally, the implementation viewpoint focuses on the technical depiction of an IoT system, and the technology and system constituents necessary for the implementation of functions and activities determined by the functional and usage viewpoints.

Regarding the framework development, Section 5.3 included a three-layered framework–edge, context information management and application–that introduced context awareness to enhance remote monitoring services in manufacturing environments. The edge layer involves the sensing and collection of context via heterogeneous sensors and integrated devices. The context information management layer includes the pre-processing of data deriving from the context-sensing layer, the interpretation of these data by the context ontology model, and the inferences drawn by the context reasoner. Finally, the interface represented by the application layer enables users to access the context information provided by the context management layer.

This developed framework allows managers and operators to assume control and make more appropriate decisions based on data, thus facilitating improved, real-time communication among managers and the shop floor. Furthermore, the proposed framework will help those companies to interpret the situation of a monitoring service and its associated data acquisition in order to identify various faults at the monitoring level and to propose actions contextually relevant to the identified situation.

RO3 - Developing a Maintenance Context Ontology for the Framework

RO3 was adopted to develop a maintenance context ontology for the framework, focusing on modelling the failure analysis of mechanical components. The aim of ontology-based context modelling is to produce a semantic organisation of data so as to drive maintenance services adaptation. When users (e.g. maintenance engineers) interact with systems in this regard, the proposed maintenance ontology can help them to narrow down the list of options. In order to achieve this objective, a maintenance ontology was developed, employing established methodologies, and through consultation of a range of domain-relevant international standards. This was presented in Chapter 5 and published in a journal. While following an established ontology development process, its design differed from other approaches in that it expanded FMEA/FMECA-based ontology constructs with additional concepts adopted from available standards in the field, which linked the key reliability-based concepts of the knowledge constructs with asset-level and fault-specific relevant diagnosability concepts.

Thus, queries could be raised, in terms of the resolution of the monitoring service context, to determine the failure mode of machinery and its potential causes, in addition to the relevant measurement parameters. Moreover, SWRL reasoning rules were used, based on ISO10816-3:2009, to evaluate the data gathered, perform the prognosis of failure, and send a maintenance message for intervention in the machine. In this way, the maintenance intervention was more directed, ceasing to be exploratory. This highlighted the need for handling the whole context information management lifecycle and ontologies in maintenance and asset management to maximise the value delivered by physical engineering assets.

RO 4 - Validating the Proposed Ontology and Architecture

The final RO involved applying the framework on a use case through an implementation architecture, and validating it through experimentation and expert judgement. The selected gearbox test rig case study was employed to validate the applicability of the proposed framework and ontology. In this regard, several ontology evaluations have been proposed, which can take an implementation or a design viewpoint (Degbelo, 2017; Kumar and Baliyan, 2018). The process of validation involved verification of the ontology's syntactical and semantic correctness, in addition to ensuring that the ontology satisfied the planned requirements. The scope of the present case study was exploratory, the aim being to present the development of a context resolution service mechanism for industrial diagnostics, based on the design of a maintenance ontology focused on modelling the failure analysis of mechanical components. Therefore, it was considered appropriate to focus on a subset of evaluation criteria–namely, robustness, level of detail, effectiveness, internal consistency and applicability–within the viewpoint of the targeted application case study. In addition to the case-study validation process, expert judgement was sought, which was generally positive. Results showed that the individual scores are significantly over the acceptable threshold of 50.9%, where the average was 70.35% (Good). Therefore, validation for usability was successfully achieved.

7.1.2 Review of the Research Questions

This section presents the RQs relevant to this exploratory study. The questions are assessed in the context of the research undertaken. A concise response to the RQs is summarised in **Table 7-1**.

Table 7-1: RQs and review informed from the research

RO	Review
RQ 1: How can context be acquired, modelled, processed and disseminated for industrial monitoring services?	This RQ was addressed in Chapter 2 by reviewing the state of the art in context lifecycle management. Context lifecycle management refers to how data is gathered, modelled and processed, and how knowledge is deduced from the obtained data. It presents how data moves from stage to stage in software systems. It also demonstrates where the data comes from and where it is consumed. In light of that, the appropriate factors used in context lifecycle management were critically analysed (Section 2.3.4).
RQ 2: What is an appropriate framework to manage context awareness in a way that facilitates efficient condition monitoring?	This RQ was addressed in Chapters 4 and 5. A framework, and an architecture that introduced context awareness to enhance remote monitoring services, were proposed. The framework applies context-aware computing to deliver solutions and address key challenges that IoT-enabled monitoring services need to handle, specifically how the context be can modelled, processed and disseminated for remote monitoring services, and how this impacts the service discovery solution. Moreover, Chapter 5 presented a maintenance context ontology for the framework, focused on the failure analysis of mechanical components, so as to drive monitoring services adaptation.
RQ 3: How can the proposed framework, which integrates of IoT and cloud computing for industrial monitoring services, be validated?	This RQ was addressed in Chapter 6. The framework was validated through a real-world case study and expert judgment. In this respect, a gear test rig was used as a case study to validate the applicability of the proposed framework and ontology. This was followed by an analysis of the evaluations obtained through expert judgment of the applicability of the framework.

7.2 Research Conclusion

- This study was aimed at developing a context-awareness framework and implementing it in an architecture that integrated the IoT with cloud computing for industrial monitoring services as a means of addressing challenges related to information complexity, scalability and data heterogeneity, as well as integrating data with domain knowledge in industrial monitoring applications. To achieve this, the problem had to be solved theoretically, by developing a framework, architecture and maintenance ontology, and validating these through laboratory experiments in a case study.
- The overall aim was achieved through a methodological investigation consisting of four phases. The first phase was focused on setting the research aim and objectives, after a thorough investigation of the research background that confirmed the validity of the research aim. The second phase was based on academic investigation of the existing literature on context-awareness frameworks, the definition and capturing of context acquisition factors, identification of a representation mechanism appropriate for context modelling, investigation of the process of context reasoning and dissemination in order to derive high-level context deductions from a set of contexts, and the delivery of context to end-user applications. The third phase consisted of experimentation materials and methods. Specifically, this involved early prototype, virtual prototype and context-model development. The final phase was focused on pulling together all the content into a framework based on context information management for industrial monitoring services.
- Regarding the framework development, the framework comprises three layers: the edge, context information management, and application. Moreover, a maintenance context ontology for the framework has developed focused on modelling failure analysis of mechanical components, so as to drive monitoring services adaptation. The developed context-awareness architecture is expressed business, usage, functional

and implementation viewpoints to frame concerns of relevant stakeholders.

- The developed framework was validated through a case study and expert judgement that provided supporting evidence for its validity and applicability in industrial contexts.
- The outcomes of the work can be used in other industrially-relevant application scenarios to drive maintenance service adaptation. Context adaptive services can help manufacturing companies in better managing the value of their assets, while ensuring that they continue to function properly over their lifecycle.

7.3 Research Limitations

Research limitations are usual and are encountered by any study, and so does this study. These limitations form the basis for further research on the same topic. The underlying objectives of the research have been achieved. As elaborated in section 1.2, the research objectives included analysing the current practices of context lifecycle management, developing a context-aware framework to enhance monitoring services, developing and validating a context resolution service mechanism for industrial diagnostics.

Some limitations in the research process, however, need to be acknowledged. One inherent limitation arose from the experimentation materials adopted, from choosing simple sensors and embedded devices for academic purposes, rather than using other methods, such as sensors and embedded devices for industrial application. In addition, only one real physical asset was used in the case study validation process; more cases could have been explored using actual physical assets. Since this study was limited to the functional issues, it was not possible to address some of the non-functional issues, such as IoT security, which constitutes a critical adoption barrier in current IIoT-enabled systems.

Context-based adaptations are not adapted in real operating conditions, but on realistic scenarios. Consequently, the outcomes of the work can be used in other industrially-relevant application scenarios to drive maintenance service

adaptation. While the application focus is quite specific, the ontology abstraction level is actually such that it could also be implemented on other application cases, as it offers a sound baseline for further customisation or extensions. When serving different application scenarios, the derived abstract model, developed using the process described in Chapter 5, still holds, as the employed terms and relationships were developed using established standards. However, after going through a similar process to derive the application-specific part of the ontology, as described in Chapter 6, and the developed queries, additional needs were identified, which may require the inclusion of additional entities, relationships and query development. This will be determined by going through an ontology assessment and evaluation cycle in the context of the new application scenario, particularly regarding the ontology expressiveness and coverage.

Moreover, while simple single-parameter threshold-based rules might be easy to interpret, they do not often hold in practice. Instead, more complex multi-parameter rules are more likely to apply. The reasoning process can replace simple rules with the activation of more complex decision functions, which may be produced as a result of machine learning over monitoring historical data. The value of the described process is that it sits at a higher level of abstraction, and can therefore work with different lower level computational rules.

Finally, whilst the aim was to capture a broad array of expert views, only 2 appear as experts and others as having professional experience. More expert judgement on the developed ontology could have been sought.

7.4 Future Work

This study has raised opportunities for further research in three areas that are explained below.

Firstly, the findings reveal a substantive theory on a topic that has not yet been studied, according to the available literature and the researcher's knowledge—a method of combining the IoT and cloud computing with context awareness for industrial monitoring services, which has not yet been covered or investigated.

Secondly, from the research limitations identified above, future work could involve further case-study validations. These could help further develop/refine the developed maintenance ontology and framework. Consequently, further research should be carried out to link the current ontology implementation with a live condition-monitoring service, as well as to apply it to real industrial environments as an enabler of more efficient IoT-enabled monitoring services. Moreover, the example reasoning rules presented in Chapter 6 could be re-used, but also could be extended with additional ones to address the coverage and expressiveness of the updated ontology.

Thirdly, as context information modelling implementation in industrial monitoring services is still in its infancy, further work could include expanding the context information management layer in the proposed framework (context modelling and context reasoning) to give appropriate information or services to consumers utilising context information, so as to drive maintenance services adaptation. Consequently, further research is required to develop context-aware approaches and architectures to deliver more efficient IoT-enabled monitoring services, including non-functional issues, such as IoT security, which constitute a critical adoption barrier in current IIoT-enabled systems.

Fourthly, and finally, one of the key issues in maintenance is to allocate focus and resources to those components and subsystems which are the most unreliable and prone to failures. In industrial systems, two methods that can be usually brought up when talking about failure identification are FMEA/FME(C)A and Fault Tree Analysis (FTA). For this research, the proposed ontology was

developed based on the FMEA technique. A further study with more focus on FTA technique is therefore suggested. FTA technique can be used as part of maintenance culture allows for more data-driven decisions..

REFERENCES

Aarnio, P., Vyatkin, V., and Hastbacka, D. (2016). “Context modeling with situation rules for industrial maintenance,” in IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA). (Berlin), 1–9. doi: 10.1109/ETFA.2016.7733539.

Abowd, G. D., and Mynatt, E. D. (2000). “Charting past, present, and future research in ubiquitous computing. ACM Trans”. Comput. Hum. Interact. 7, 29–58. doi: 10.1145/344949. 344988.

Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P. (1999). “Towards a better understanding of context and context-awareness,” in Proceeeding of 1st International Symposium on Handheld and Ubiquitous Computing, ser. HUC ’99. (London, UK: Springer-Verlag), 304–307. doi: 10.1007/3-540-48157-5_29.

Aguilar, J., Jerez, M. and Rodríguez, T. (2018) “CAMeOnto: Context awareness meta ontology modelling”, Applied Computing and Informatics. King Saud University, 14(2), pp. 202–213. doi:org/10.1016/j.aci.2017.08.001.

AIOTI (2016) High Level Architecture- Release 2.1 AIOTI WG03 – loT Standardisation. doi: 10.1177/003754979907300501.

AIOTI (2018) High Level Architecture - AIOTI WG03 – loT Standardisation. doi: 10.1177/003754979907300501.

AIOTI (2019) IoT LSP Standard Framework Concepts- Release 2.9 AIOTI WG03 – loT Standardisation.

AIOTI, (2015): Internet of Things Applications AIOTI WG01–IERC

Alegre, U., Augusto, J. C., and Clark, T. (2016). Engineering context-aware systems and applications: a survey. J. Syst. Softw. 117, 55–83. doi: 10.1016/j.jss.2016.02.010.

Al-Fuqaha, Ala. Guizani, Mohsen. Mohammadi, Mehdi. Aledhari, Mohammed. Ayyash, M. (2015). "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," in IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347-2376, Fourth quarter 2015, doi: 10.1109/COMST.2015.2444095.

Al-shdifat, A. and Emmanouilidis, C. (2018) 'Development of a Context-aware framework for the Integration of Internet of Things and Cloud Computing for Remote Monitoring Services', Procedia Manufacturing. Elsevier B.V., 16, pp. 31–38. doi: 10.1016/j.promfg.2018.10.155.

Al-Shdifat A., Emmanouilidis C., Starr A. (2020) "Context-Awareness in Internet of Things Enabled Monitoring Services". In: (eds) Engineering Assets and Public Infrastructures in the Age of Digitalization. Liyanage J., Amadi-Echendu J., Mathew J. Lecture Notes in Mechanical Engineering. Springer, Cham. https://doi.org/10.1007/978-3-030-48021-9_98.

Atzori, L., Iera, A. and Morabito, G. (2010) 'The Internet of Things: A survey', Computer Networks, 54(15), pp. 2787–2805. doi.org/10.1016/j.comnet.2010.05.010.

Bandyopadhyay, S., Sengupta, M., Maiti, S. and Dutta, S. (2011) 'Role of Middleware for Internet of Things: a Study', International Journal of Computer Science & Engineering Survey (IJCSES), 2(3), pp. 94–105. DOI: 10.5121/ijcses.2011.2307.

Bangor, A., Kortum, P. and Miller, J. (2009) 'Determining what individual SUS scores mean', Journal of Usability Studies, 4(3), pp. 114–123.

Barnaghi, P., Wang, W., Henson, C. and Taylor, K. (2012) 'Semantics for the Internet of Things: Early Progress and Back to the Future', International Journal on Semantic Web and Information Systems, 8(1), pp. 1–21. doi.org/10.4018/jswis.2012010101.

Bernardos, A. M., Tarrío, P., and Casar, J. R. (2008). "A data fusion framework for context-aware mobile services," in Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, (South Korea: Seoul), 606–613. doi: 10.1109/MFI.2008.4648011.

Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., and Riboni, D. (2010). A survey of context modelling and reasoning techniques. Pervasive Mobile Comput. 6, 161–180. doi: 10.1016/j.pmcj.2009.06.002.

Bikakis A., Patkos T., Antoniou G., Plexousakis D. (2008) A Survey of Semantics-Based Approaches for Context Reasoning in Ambient Intelligence.

In: Constructing Ambient Intelligence. Mühlhäuser M., Ferscha A., Aitenbichler E. (eds). AmI 2007. Communications in Computer and Information Science, vol 11. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-85379-4_3.

Boavida, F., Kliem, A., Renner, T., Riekki, J., Jouvray, C., Jacovi, M., Ivanov, S., Guadagni, F., Gil, P., & Triviño-Cabrera, A. (2016) People-Centric Internet of Things—Challenges, Approach, and Enabling Technologies. In: Novais P., Camacho D., Analide C., El Fallah Seghrouchni A., Badica C. (eds) Intelligent Distributed Computing IX. Studies in Computational Intelligence, vol 616, 463–474. Springer, Cham. https://doi.org/10.1007/978-3-319-25017-5_44.

Botta, A., De Donato, W., Persico, V. and Pescapé, A. (2016) 'Integration of Cloud computing and Internet of Things: A survey', Future Generation Computer Systems, 56 Elsevier B.V., pp. 684–700. doi.org/10.1016/j.future.2015.09.021.

Bousdekis, A., Magoutas, B., Apostolou, D. and Mentzas, G. (2015), "A proactive decision making framework for condition-based maintenance", Industrial Management & Data Systems, Vol. 115 No. 7, pp. 1225-1250. DOI: 10.1108/IMDS-03-2015-0071.

Brooke, J. (1996). SUS: a „quick and dirty" usability scale. In P.W.Jordan, B. Thomas, B.A. Weerdmeester, and I.L. McClelland (Eds.) Usability Evaluation in Industry (189-194). London: Taylor and Francis.

Bryman, A. and Bell, E. (2011) 'Business Research Methods'. 3rd edition. Oxford: Oxford University Press.

Bryman, A. and Bell, E. (2015) 'Business Research Methods'. 4th edition. Oxford: Oxford University Press.

BS ISO 13306, (2010) 'BSI Standards Publication Maintenance — Maintenance terminology'. Vocabulary, 2010.

BS ISO 13373-2:2016 "Condition monitoring and diagnostics of machines. Vibration condition monitoring. Processing, analysis and presentation of vibration data"

BS (2012) 'BS ISO 13373-2:2016 - Condition monitoring and diagnostics of machines — Data processing , communication and presentation', 3.

Burhan, M., Rehman, R.A., Khan, B. and Kim, B.S. (2018) 'IoT elements, layered architectures and security issues: A comprehensive survey', Sensors (Switzerland), 18(9), pp. 1–37. doi:10.3390/s18092796.

Burrell, G. and Morgan, G. (1979) Sociological paradigms and organisational analysis: elements of the sociology of corporate life. 14th edition. England: Heinemann Educational Books. Business Students. 6th edn. Pearson Education.

Byun, H. E., and Cheverst, K. (2004) Utilizing context history to provide dynamic adaptations. Appl. Artif. Intel. 18, 533–548. doi: 10.1080/08839510490462894.

Cabrera, O., Franch, X. and Marco, J. (2017) '3LConOnt: a three-level ontology for context modelling in context-aware computing', Software and Systems Modeling. Springer Berlin Heidelberg, pp. 1–34. doi.org/10.1007/s10270-017-0611-z.

Cândea, G., Kifor, S. and Constantinescu, C. (2014), "Usage of Case-based Reasoning in FMEA-driven Software", Procedia CIRP, Vol. 25 No. 0, pp. 93-99. doi.org/10.1016/j.procir.2014.10.016.

Cao, Q., Samet, A., Zanni-Merk, C., De Beuvron, F. D. B., and Reich, C. (2019). An ontology-based approach for failure classification in predictive maintenance using fuzzy c-means and SWRL rules. Procedia Comput. Sci. 159, 630–639. doi: 10.1016/j.procs.2019.09.218**.**

Casellas N. (2009) 'Ontology Evaluation through Usability Measures'. In: Meersman R., Herrero P., Dillon T. (eds) On the Move to Meaningful Internet Systems: OTM 2009 Workshops. OTM 2009. Lecture Notes in Computer Science, vol 5872. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-05290-3_73.

Castet, J. F., Bareh, M., Nunes, J., Okon, S., Garner, L., Chacko, E., and Izygon, M. (2018). "Failure analysis and products in a model-based environment," in IEEE Aerospace Conference Proceedings (IEEE) (Big Sky, MT), 1–13. doi: 10.1109/AERO.2018.8396736.

Chen, G., and Kotz, D. (2000). A Survey of Context-Aware Mobile Computing Research. Technical Report. Hanover, NH: Department of Computer Science,

Dartmouth College, 1–16. Available online at: http://www.cs.dartmouth.edu/ reports/TR2000- 381.pdf (accessed June 16, 2020).

Chen, H., Finin, T., Joshi, A., Kagal, L., Perich, F., and Chakraborty, D. (2004). Meet the semantic web in smart spaces. IEEE Internet Comput. 8, 69–79. doi: 10.1109/MIC.2004.66.

Chong, S. K., McCauley, I., Loke, S. W., and Krishnaswamy, S. (2007). "Context-aware sensors and data muling," in Context Awareness for Self-Managing Systems (Devices, Applications and Networks) Proceeding (Berlin: VDE-Verlag), 103–117.

CISCO, 2019. "Demystifying 5G in Industrial IOT" [White paper]. Available at: https://www.cisco.com/c/dam/en_us/solutions/iot/demystifying-5g-industrial-iot.pdf (Accessed: 12, October 2020).

Contreras, J. D., Garcia, J. I. and Pastrana, J. D. (2017) 'Developing of industry 4.0 applications', International Journal of Online Engineering, 13(10), pp. 30–47.

Collins, V. and Lanz, J. (2019) "Managing Data as an Asset". Available online at:https://www.cpajournal.com/2019/06/24/managing-data-as-an asset/#:~:text=Data%20is%20an%20economic%20asset,allow%20innovation%2C%20and%20reduce%20risks. (accessed July 2019).

Creswell, J. W. (2009). Research Design: Qualitative, Quantitative, and Mixed Method Approaches. SAGE.

Creswell, J. W., & Poth, C. N. (2016). Qualitative inquiry and research design: Choosing among five approaches. London: Sage Publications. doi:10.1086/317417.

D'Elia, A., Roffia, L., Zamagni, G., Vergari, F., Toninelli, A., & Bellavista, P. (2010). Smart applications for the maintenance of large buildings: How to achieve ontology-based interoperability at the information level. IEEE Symposium on Computers and Communications, 1077–1082. https://doi.org/10.1109/ISCC.2010.5546633.

da Silva, M. J., Pereira, C. E., and Götz, M. (2018). Context-aware recommendation for industrial alarm system. IFAC-PapersOnLine 51, 229–234. doi: 10.1016/j.ifacol.2018.08.266.

de Matos, E., Amaral, L. A., and Hessel, F. (2017). Context-Aware Systems: Technologies and Challenges in Internet of Everything Environments. Cham: Springer International Publishing, 1–25. doi: 10.1007/978-3-319-507 58-3_1.

de Matos, E., Tiburski, R. T., Moratelli, C. R., Johann Filho, S., Amaral, L. A., Ramachandran, G., Krishnamachari, B., and Hessel, F. (2020). Context information sharing for the internet of things: a survey. Computer Networks. 166:106988. doi: 10.1016/j.comnet.2019.106988.

de Rocha, R. C. A., and Endler, M. (2006). “Middleware: context management in heterogeneous, evolving ubiquitous environments,” in IEEE Distributed Systems Online (IEEE), 1–13. doi: 10.1109/MDSO.2006.28.

del Castillo, A.C. Diebolt, P. Hazi, C. Kreinin, M. Li, w. (2020) " Industrial Internet of Things for Monitoring Services". Group Project at School of Aerospace, Transport and Manufacturing MSM/MTM at Cranfield Univesity.

Degbelo, A. (2017). A Snapshot of Ontology Evaluation Criteria and Strategies. Semantics2017. (pp. 1–8). doi.org/10.1145/3132218.3132219.

Delicato F.C., Pires P.F., Batista T. (2017) The Resource Management Challenge in IoT. In: Resource Management for Internet of Things. SpringerBriefs in Computer Science. Springer, Cham. https://doi.org/10.1007/978-3-319-54247-8_2.

Dey, A., Abowd, G., and Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Hum. Comput. Interact. 16, 97–166. doi: 10.1207/S15327051HCI16 234_02.

Dhallenne, J., Jayaraman, P. P., and Zaslavsky, A. (2016). RCOS: Real Time Context Sharing Across a Fleet of Smart Mobile Devices. Cham: Springer International Publishing, 87–100. doi: 10.1007/978-3-319-46301-8_8.

Dimitrova, V., Mehmood, M. O., Thakker, D., Sage-Vallier, B., Valdes, J., and Cohn, A. G. (2020). An ontological approach for pathology assessment and diagnosis of tunnels. Eng. Appl. Artif. Intellig. 90:103450. doi: 10.1016/j.engappai.2019.103450.

Dibley, M. Li, H.J. Rezgui, Y. Miles, J.C. (2012) Cost-effective and scalable sensor network for intellient building monitoring, International Journal of Innovative Computing, Information and Control 8 (12).

Domingo, M.C. (2012) ‘An overview of the Internet of Things for people with disabilities’, Journal of Network and Computer Applications, 35(2) Elsevier, pp. 584–596. doi.org/10.1016/j.jnca.2011.10.015.

Doukas, C., Capra, L., Antonelli, F., Jaupaj, E., Tamilin, A., and Carreras, I. (2015). “Providing generic support for IoT and M2M for mobile devices,” in: The 2015 IEEE RIVF International Conference on Computing Communication Technologies—Research, Innovation, and Vision for Future (RIVF) (Can Tho: IEEE), 192–197. doi: 10.1109/RIVF.2015.70 49898.

Du, J., Wang, S. and Zhang, H. (2013) ‘Layered clustering multi-fault diagnosis for hydraulic piston pump’, Mechanical Systems and Signal Processing. Elsevier, 36(2), pp. 487–504. doi.org/10.1016/j.ymssp.2012.10.020.

Ebrahimipour, V., and Yacout, S. (2015). “Ontology-based schema to support maintenance knowledge representation with a case study of a pneumatic valve,” in IEEE Transactions on Systems, Man, and Cybernetics: Systems (IEEE), 702–712. doi: 10.1109/TSMC.2014.2383361.

El Kadiri, S., Grabot, B., Thoben, K. D., Hribernik, K., Emmanouilidis, C., Von Cieminski, G., & Kiritsis, D (2016), Current trends on ICT technologies for enterprise information systems. Computers in Industry, 79, 14–33.

Emmanouilidis, C., Koutsiamanis, R. A., and Tasidou, A. (2013). Mobile guides: taxonomy of architectures, context awareness, technologies and applications. J. Netw. Comput. Appl. 36, 103–125. doi: 10.1016/j.jnca.2012.04.007.

Emmanouilidis, C., Pistofidis, P., Bertoncelj, L., Katsouros, V., Fournaris, A., Koulamas, C., Ruiz-Carcela, C. (2019). Enabling the human in the loop: linked data and knowledge in industrial cyber-physical systems. Annu. Rev. Control. 47, 249–265. doi: 10.1016/j.arcontrol.2019.03.004.

Farghaly, K., Abanda, F. H., Vidalakis, C., and Wood, G. (2019). BIM-linked data integration for asset management. Built Environ. Project Asset Manage. 9, 489–502. doi: 10.1108/BEPAM-11-2018-0136.

Farrar, CR. Worden. K. (2012) 'Structural Health Monitoring: A Machine Learning Perspective'. Structural Health Monitoring: A Machine Learning Perspective. ISBN: 978-1-118-44321-7

France-Mensah, J., and O'Brien, W. J. (2019). A shared ontology for integrated highway planning. Adv. Eng. Inform. 41:100929. doi: 10.1016/j.aei.2019.100929.

Friess, P. and Vermesan, O. (2015) Building the Hyperconnected Society - Internet of Things Research and Innovation Value Chains, Ecosystems and Markets. Denmark: River publisher. doi: 978-87-93237-99-5.

Frolov V., Mengel D., Bandara W., Sun Y., Ma L. (2010) Building an ontology and process architecture for engineering asset management. In: Kiritsis D., Emmanouilidis C., Koronios A., Mathew J. (eds) Engineering Asset Lifecycle Management. Springer, London. https://doi.org/10.1007/978-0-85729-320-6_11.

Garlan, D., Siewiorek, D. P., Smailagic, A., and Steenkiste, P. (2002). 'Project aura: toward distraction-free pervasive computing. IEEE Pervasive Comput. 1, 22–31. doi: 10.1109/MPRV.2002.1012334.

Gil, D., Ferrández, A., Mora-Mora, H. and Peral, J. (2016) 'Internet of things: A review of surveys based on context aware intelligent services', Sensors (Switzerland), 16(7), pp. 1–23. doi: 10.3390/s16071069.

Girdhar, P. and Scheffer, C. (2004) 'Machinery fault diagnosis using vibration analysis', Practical Machinery Vibration Analysis and Predictive Maintenance, pp. 89–133.

Gregori, M (2018), 'Context Information Management for Connected Services', Thesis, School of Aerospace, Transport and Manufacturing Management and Information Systems, Cranfield University, UK.

Hammar, K. (2017), 'Content Ontology Design Patterns Qualities, Methods, and Tools', Thesis, Department of Computer and Information Science, Linköping University, Sweden.

Hassani, A., Medvedev, A., Haghighi, P. D., Ling, S., Indrawan-Santiago, M., Zaslavsky, A., and Jayaraman, P. (2018). "Context-as-a-service platform: exchange and share context in an IoT ecosystem," in 2018 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops (Athens: IEEE), 385–390. doi: 10.1109/PERCOMW.2018.8480240.

Henricksen, K. (2003). A framework for context-aware pervasive computing applications. (Ph. D thesis). The School of Information Technology and Electrical Engineering, The University of Queensland, 13–20.

Herbert, J., O'donoghue, J. and Chen, X. (2008) 'A context-sensitive rule-based architecture for a smart building environment'. Second International Conference on Future Generation Communication and Networking, Hainan Island, 2008, pp. 437-440, doi: 10.1109/FGCN.2008.169.

Hoffmann, J.B., Heimes, P. and Senel, S. (2019) 'IoT platforms for the internet of production', IEEE Internet of Things Journal, 6(3), pp. 4098–4105.

Hodkiewicz, M., Klüwer, J.W., Woods, C., Smoker, T. and French, T. (2020) 'Digitalization and reasoning over engineering textual data stored in spreadsheet tables', IFAC-PapersOnLine, 53(3) Elsevier Ltd, pp. 239–244.

Hynes, G., Reynolds, V., and Hauswirth, M, (2009) "A context lifecycle for web-based context management services," in Smart Sensing and Context, ser. Lecture Notes in Computer Science, P. Barnaghi, K. Moessner, M. Presser, and S. Meissner, Eds. Springer Berlin/ Heidelberg, vol. 5741, pp. 51–65.

IIC, Industrial Internet Consortium. (2017) "the Industrial Internet of Things Volume G1: Reference Architecture". IIC:PUB:G1:V1.80:20170131. Available at: https://www.iiconsortium.org/IIRA.htm. (Accessed: 17 January 2019).

IEC 60812:2018 Failure modes and effects analysis (FMEA and FMECA).

Indulska, J. and Sutton, P. (2003) 'Location management in pervasive systems', ACSW Frontiers '03 Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003, 21, pp. 143–151. doi: 10.1145/830000/828003.

Ismail, A.A., Hamza, H.S. and Kotb, A.M. (2018) 'Performance Evaluation of Open Source IoT Platforms', 2018 IEEE Global Conference on Internet of Things, GCIoT 2018, IEEE.

ISO 10816-3:2009 Mechanical vibration — Evaluation of machine vibration by measurements on non-rotating parts.

ISO 13372, (2012) 'International Organization for Standardization', in Condition monitoring and diagnostics of machines – Vocabulary, 2012.

ISO 13372. (2012). Condition monitoring and diagnostics of machines — Vocabulary.

ISO 13373-1:2002 Condition monitoring and diagnostics of machines — Vibration condition monitoring.

ISO 13373-2:2016 Condition monitoring and diagnostics of machines — Vibration condition monitoring.

ISO 13374:1. (2003). Condition Monitoring and diagnostics of machines - Data processing, communication and presentation - Part 1: general guidelines.

ISO 13379-1 (2012) 'BSI Standards Publication Condition monitoring and diagnostics of machines -Data processing, communication and presentation'.

ISO 17359. (2011). Condition monitoring and diagnostics of machines: General guidelines.

ISO 2041, (2009) 'international Organization for Standardization', in Mechanical vibration, shock and condition monitoring – Vocabulary, 2009.

ISO 2041. (2018). Mechanical vibration, schock and condition monitoring - vocabulary.

ISO 2041. (2018). Mechanical vibration, shock and condition monitoring - vocabulary.

ISO 55000, (2014) 'International Organization For Standardization', in Asset management – Overview, principles and terminology, 2014.

ISO 55000. (2014). Asset management — Overview, principles and terminology

ISO/IEC/IEEE: "ISO/IEC/IEEE 42010:2011 Systems and software engineering - Architecture description", 2011. Available at: http://www.iso.org/iso/catalogue_detail.htm?csnumber=50508 . (Accessed: 20 January 2020).

ISO 13306. (2017). Maintenance - Maintenance Terminology.

Issarny, V., Caporuscio, M., Georgantas, N., Issarny, V. and Georgantas, N. (2002) 'A Perspective on the Future of Middleware-based Software Engineering. Future of Software Engineering (FOSE '07), Minneapolis, MN, 2007, pp. 244-258, doi: 10.1109/FOSE.2007.2.

Jeschke S., Brecher C., Meisen T., zdemir D., and Eschert T. (2017). "Industrial internet of things and cyber manufacturing systems," in Industrial Internet of

Things. Springer Series in Wireless Technology, eds S. Jeschke, C. Brecher, H. Song, and D. Rawat. Pp. 3-20 (Cham: Springer). doi: 10.1007/978-3-319-42559-7_1.

Jia, X., Feng, Q., Fan, T. and Lei, Q. (2012) 'RFID technology and its applications in Internet of Things (IoT)', 2012 2nd International Conference on Consumer Electronics, Communications and Networks, CECNet 2012 - Proceedings, (July), pp. 1282–1285.

Jih, W., Huang, C. and Hsu, J. (2009) 'Context Life Cycle Management in Smart Space Environments', Proceedings of the 3rd workshop on Agent- …, pp. 9–14.

Jin, G., Xiang, Z., & Lv, F. (2009). Semantic integrated condition monitoring and maintenance of complex system. 16th International Conference on Industrial Engineering and Engineering Management, Beijing, China, pp. 670-674, doi: 10.1109/ICIEEM.2009.5344503.

Karray, M. H., Chebel-Morello, B., and Zerhouni, N. (2012). A formal ontology for industrial maintenance. Appl. Ontol. 7, 269–310. doi: 10.3233/AO-2012-0112.

Keivanpour, S., and Ait Kadi, D. (2019). Internet of things enabled real-time sustainable end-of-life product recovery. IFAC PapersOnLine 52, 796–801. doi: 10.1016/j.ifacol.2019.11.213.

Kessler, C. Raubal, M. and Wosniok, C. 2009 "Semantic rules for context-aware geographical information retrieval," in Proc. 4th European conference on Smart sensing and context, ser. EuroSSC'09. Berlin, Heidelberg: Springer-Verlag.

Khan, M.A., Shahid, M.A., Ahmed, S.A., Khan, S.Z., Khan, K.A., Ali, S.A. and Tariq, M. (2019) 'Gear misalignment diagnosis using statistical features of vibration and airborne sound spectrums', Measurement: Journal of the International Measurement Confederation, 145 Elsevier Ltd, pp. 419–435. doi: 10.1016/j.measurement.2019.05.088.

Klaauw, T. Van Der (2019) IoT Platforms for Cities : a Comparative Survey. The Academy for Smarter Communities.

Koukias, A., Nadoveza, D., and Kiritsis, D. (2013). Semantic data model for operation and maintenance of the engineering asset. IFIP Adv. Inform. Commun. Technol. 398, 49–55. doi: 10.1007/978-3-642-40361-3_7.

Kozaki, K. Kitamura, Y. Mizoguchi R. (2005) Developing ontology-based applications using Hozo, Computational Intelligence, A Scientific and Technical Publishing Company, 2005.

Kumar, R. (2005). Research methodology – A step-by-step guide for beginners. SAG.

Kumar, S., & Baliyan, N. (2018). Quality evaluation of ontologies. Semantic Web-Based Systems: Quality Assessment Models (pp. 19–50).

Lawan, A., and Rakib, A. (2019). The semantic web rule language expressiveness extensions-a survey. arXiv 11723.

Lee, A. N. and Martinez Lastra, J. L. (2013) "Enhancement of industrial monitoring systems by utilizing context awareness," 2013 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), San Diego, CA, USA, 2013, pp. 277-284, doi: 10.1109/CogSIMA.2013.6523858.

Li, R., Mo, T., Yang, J., Jiang, S., Li, T., and Liu, Y. (2020). “Ontologies-based domain knowledge modeling and heterogeneous sensor data integration for bridge health monitoring systems,” in IEEE Transactions on Industrial Informatics (IEEE), 3203. doi: 10.1109/TII.2020.2967561.

Li, X., Eckert, M., Martinez, J. F. and Rubio, G. (2015) ‘Context aware middleware architectures: Survey and challenges’, Sensors (Switzerland), 15(8), pp. 20570–20607.

Liu, C., Hua, J., and Julien, C. (2019). “SCENTS: collaborative sensing in proximity IoT networks,” in 2019 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops (Kyoto: IEEE), 189–195. doi: 10.1109/PERCOMW.2019.8730863.

Liu, C.H., Yang, B. and Liu, T. (2014) ‘Efficient naming, addressing and profile services in Internet-of-Things sensory environments’, Ad Hoc Networks, 18 Elsevier B.V., pp. 85–101.

Liu, W., Li, X., and Huang, D. (2011). “A survey on context awareness,” in 2011 International Conference on Computer Science and Service System (CSSS) (Nanjing: IEEE), 144–147. doi: 10.1109/CSSS.2011.5972040.

Lunardi, W. T., de Matos, E., Tiburski, R., Amaral, L. A., Marczak, S., and Hessel, F. (2015). “Context-based search engine for industrial IoT: discovery, search, selection, and usage of devices,” in 2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA), (Luxembourg), 1–8. doi: 10.1109/ETFA.2015.7301477.

Matsokis, A., and Kiritsis, D. (2012). “Ontology-based implementation of an advanced method for time treatment in asset lifecycle management,” in: Engineering Asset Management and Infrastructure Sustainability. eds J. Mathew, L. Ma, A. Tan, M. Weijnen, J. Lee (London, UK: Springer), 647–662. doi: 10.1007/978-0-85729-493-7_50.

Mell, P. and Grance, T. (2011) The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology, Nist Special Publication.

MIMOSA. (2010) Open System Architecture for Enterprise Application Integration V3.3.1, Available at: http://www.mimosa.org/mimosa-osa-cbm/, (Accessed: 20 February 2018).

Mishra, S.B. Alok, S. (2011). "Handbook of Research Methodology, a compendium for scholars & researchers". EDUCATION PUBLISHING. Dwarka, New Delhi - 110075

Mineraud, J., Mazhelis, O., Su, X., and Tarkoma, S. (2016). A gap analysis of Internet-of-Things platforms. Comput. Commun. 89–90, 5–16. doi: 10.1016/j.comcom.2016.03.015.

Monnin, M., Abichou, B., Voisin, A. and Mozzati, C. (2011) ‘Fleet historical cases for predictive maintenance’, Applied Soft Computing, 69(May 2014), pp. 213–231.

Ngu, A.H., Gutierrez, M., Metsis, V., Nepal, S. and Sheng, Q.Z. (2017) ‘IoT Middleware: A Survey on Issues and Enabling Technologies’, IEEE Internet of Things Journal, 4(1) IEEE, pp. 1–20.

Noy, N. F., and McGuinness, D. L. (2001). A guide to creating your first ontology. Biomed. Inform. Res. 7–25. Available online at: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf (accessed June 15, 2020).

Nunez, D. L., and Borsato, M. (2018). OntoProg: an ontology-based model for implementing prognostics health management in mechanical machines. Adv. Eng. Inform. 38, 746–759. doi: 10.1016/j.aei.2018.10.006.

Ong, S. K., and Zhu, J. (2013). A novel maintenance system for equipment serviceability improvement. CIRP Ann. Manuf. Technol. 62, 39–42. doi: 10.1016/j.cirp.2013.03.091.

Papadopoulos, P., and Cipcigan, L. (2010). "Wind turbines condition monitoring: an ontology model," in International Conference on Sustainable Power Generation and Supply (Nanjing: IEEE), 1–4. doi: 10.1109/SUPERGEN.2009.5430854.

Peng, W. (2011) 'Remote Online Machine Condition Monitoring Using Advanced Internet, Wireless and Mobile Communication Technologies', Thesis, (July), p. 10.

Perera, C., Liu, C. H., Jayawardena, S., and Chen, M. (2015). A survey on internet of things from industrial market perspective. IEEE Access 2, 1660–1679. doi: 10.1109/ACCESS.2015.2389854.

Perera, C., Zaslavsky, A., Christen, P., and Georgakopoulos, D. (2014). Context aware computing for the internet of things. IEEE Commun. Surveys Tutorials 16, 414–454. doi: 10.1109/SURV.2013.042313.00197.

Perttunen, M., Riekki, J., and Lassila, O. (2009). Context representation and reasoning in pervasive computing: a review. Int. J. Multimedia Ubiquit. Eng. 4, 1–28.

Pietschmann, S., Mitschick, A., Winkler, R. and Meißner, K. (2008) 'CROCO: Ontology-based, cross-application context management', Proceedings - 3rd International Workshop on Semantic Media Adaptation and Personalization, SMAP 2008, pp. 88–93.

Pradeep, P., and Krishnamoorthy, S. (2019). The MOM of context-aware systems: a survey. Comput. Commun. 137, 44–69. doi: 10.1016/j.comcom.2019.02.002.

Qin, Y., Sheng, Q. Z., Falkner, N. J., Dustdar, S. , Wang, H. , & Vasilakos, A. V. (2016). When things matter: A survey on data-centric internet of things. Journal of Net¬work and Computer Applications,64 , 137–153 . https://doi.org/10.1016/j.jnca.2015.12.016.

Raggett, D. (2015) "The Web of Things: Challenges and Opportunities," in Computer, vol. 48, no. 5, pp. 26-32, doi: 10.1109/MC.2015.149.

Ramachandran, G. S., and Krishnamachari, B. (2019). Towards a large scale IoT through partnership, incentive, and services: a vision, architecture, and future directions. Open J. Int. Things 5, 80–92**.**

Ren, G., Ding, R., and Li, H. (2019). Building an ontological knowledgebase for bridge maintenance. Adv. Eng. Softw. 130, 24–40. doi: 10.1016/j.advengsoft.2019.02.001.

Rizou, S., Haussermann, K., Durr, F., Cipriani, N., and Rothermel,. K. (2010). “A system for distributed context reasoning,” in Sixth International Conference on Autonomic and Autonomous Systems (ICAS) (Cancun: IEEE), 84–89. doi: 10.1109/ICAS.2010.21.

Ruta, M., Scioscia, F., Ieva, S., Loseto, G., Gramegna, F., and Pinto, A., (2017). “Knowledge discovery and sharing in the IoT: the physical semantic web vision,” in Proceedings of the ACM Symposium on Applied Computing (Marrakech), 492–498. doi: 10.1145/3019612.3019701.

Sanchez, L., Lanza, J., Olsen, R., Bauer, M. and Girod-Genet, M. (2006) ‘A generic context management framework for personal networking environments’, 2006 3rd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, MobiQuitous.

Saunders, M., Lewis, P. and Thornhill, A. (2018) Chapter 4: Understanding research philosophy and approaches to theory development. Eighth Edition, PP. 128-171, New York, ISBN 9781292208794 (epub).

Saunders, M., Thornhill, A. and Lewis, P. (2009) Research Methods for Business Students. 5th edition. London: Pearson Education.

Saunders, M., Thornhill, A. and Lewis, P. (2012) Research Methods for 225.

Schilit, B. N., and Theimer, M. M. (1994). “Disseminating active map information to mobile hosts.” IEEE Netw. 8, 22–32. doi: 10.1109/65.313011.

Scholze, Sebastian; Barata, Jose; Stokic, Dragan. 2017. "Holistic Context-Sensitivity for Run-Time Optimization of Flexible Manufacturing Systems" Sensors 17, no. 3: 455. https://doi.org/10.3390/s17030455.

Sezer, O. B., Dogdu, E., and Ozbayoglu, A. M. (2018). Context-aware computing, learning, and big data in internet of things: a survey. IEEE Internet Things J. 5, 1–27. doi: 10.1109/JIOT.2017.2773600.

Shi-Wan Lin, Brett Murphy, Erich Clauer, Ulrich Loewen, Ralf Neubert, Gerd Bachmann, Madhusudan Pai, M.H. (2018) Architecture Alignment and Interoperability An Industrial Internet Consortium and Plattform Industrie 4.0 Joint Whitepaper.

Sisinni, E., Saifullah, A., Han, S., Jennehag, U. and Gidlund, M. (2018) ‘Industrial internet of things: Challenges, opportunities, and directions’, IEEE Transactions on Industrial Informatics, 14(11) IEEE, pp. 4724–4734.

Snidaro, L., Garcia, J., and Llinas, J. (2015). Context-based information fusion: a survey and discussion. Inform. Fusion 25, 16–31. doi: 10.1016/j.inffus.2015.01.002.

Statista report. (2020). Number of Internet of Things (IoT) Connected Devices Worldwide in 2018, 2025 and 2030. Available online at: https://www.statista.com/statistics/802690/worldwide-connected-devices-by-access-technology/ (accessed June 22, 2020).

Strang, T. and Linnhoff-Popien, C. (2004) ‘A Context Modeling Survey’, Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Workshop o(4), pp. 1–8.

Sure Y., Staab S., Studer R. (2009) Ontology Engineering Methodology. In: Staab S., Studer R. (eds) Handbook on Ontologies. International Handbooks on Information Systems. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-92673-3_6.

Sure, Y. Angele, J. Staab S. (2002) OntoEdit: guiding ontology development by metho¬dology and inferencing, On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE, Springer, 2002, pp. 1205–1222.

Tan, H., Adlemo, A., Tarasov, V. and Johansson, M.E. (2017) 'Evaluation of an application ontology', CEUR Workshop Proceedings, 2050.

Tran, M.-Q., Nguyen, D.T., Le, V.A., Nguyen, D.H., Pham, T.V., 2019. Task placement on fog computing made efficient for iot application provision. Wireless Commun. Mobile Comput. 2019, 1–17.

Teoh, P. C. and Case, K. (2004), "Failure modes and effects analysis through knowledge modelling", Journal of Materials Processing Technology, Vol. 153–154, pp. 253-260.

Tiburski, R. T., Amaral, L. A., Matos, E. D., Hessel, F. (2015). The importance of a standard security architecture for SOA-based IoT middleware. IEEE Commun. Mag. 53, 20–26. doi: 10.1109/MCOM.2015.7355580.

Uhlmann, E., Laghmouchi, A., Hohwieler, E. and Geisert, C. (2015) 'Condition Monitoring in the Cloud', Procedia CIRP, 38, pp. 53–57. doi: 10.1016/j.procir.2015.08.075.

Uschold M, Gruninger M., (1996) "Ontologies: principles, methods and applications". Knowl Eng Rev; 11:93–136.

Uviase, O. and Kotonya, G. (2018) 'IoT Architectural Framework: Connection and Integration Framework for IoT Systems', Electronic Proceedings in Theoretical Computer Science, 264, pp. 1–17.

Valverde-Rebaza, J. C., Roche, M., Poncelet, P., and de Lopes, A. (2018). The role of location and social strength for friendship prediction in location-based social networks. Inform. Process. Manage. 54, 475–489. doi: 10.1016/j.ipm.2018. 02.004.

Van Bunningen, A. H., Feng, L., Apers, P. M. G. (2005). "Context for ubiquitous data management," in Proceedings of the International Workshop on Ubiquitous Data Management, (Washington, DC), 17–24. doi: 10.1109/UDM. 2005.7.

Veiga, E. F., Arruda, M. F., Neto, J. A. B. and Bulcão-Neto, R. de F. (2017) 'An Ontology-based Representation Service of Context Information for the Internet

of Things', Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web - WebMedia '17, pp. 301–308.

Wang, Y., Anokhin, O. and Anderl, R. (2017) 'Concept and use Case Driven Approach for Mapping IT Security Requirements on System Assets and Processes in Industrie 4.0', Procedia CIRP, 63 The Author(s), pp. 207–212.

Wei, L., Du, H., Mahesar, Q. ain., Al Ammari, K., Magee, D. R., Clarke, B., Dimitrova, V., Gunn, D., Entwisle, D., Reeves, H., and Cohn, A. (2020). A decision support system for urban infrastructure inter-asset management employing domain ontologies and qualitative uncertainty-based reasoning. Expert Syst. Appl. 158:113461. doi: 10.1016/j.eswa.2020. 113461.

Xu, L. D., He, W., and Li, S. (2014). Internet of things in industries: a survey. IEEE Trans. Ind. Inform. 10, 2233–2243. doi: 10.1109/TII.2014.2300753.

Yamamoto, J., Nakagawa, H., Nakayama, K., Tahara, Y., and Ohsuga, A. (2009). "A context sharing message broker architecture to enhance interoperability in changeable environments," in 3rd International Conference on Mobile Ubiquitous Computing, Systems, Services, and Technologies (Sliema: IEEE), 31–39. doi: 10.1109/UBICOMM.2009.48.

Yanwei, S., Guangzhou, Z. and Haitao, P. (2011) 'Research on the context model of intelligent interaction system in the Internet of Things', IT in Medicine and Education (ITME), 2011 International Symposium on, 2, pp. 379–382.

Yao, L. Sheng, Q. Z. and Dustdar, S. (2015) "Web-based management of the Internet of Things," IEEE Internet Comput., vol. 19, no. 4, pp. 60–67. DOI Bookmark: 10.1109/MIC.2015.77.

Yu, J.Y. and Kim, Y.G. (2019) 'Analysis of IoT Platform Security: A Survey', 2019 International Conference on Platform Technology and Service, PlatCon 2019 - Proceedings, IEEE.

Zhong, C. Le, Zhu, Z. and Huang, R. G. (2016) 'Study on the IOT architecture and gateway technology', Proceedings - 14th International Symposium on Distributed Computing and Applications for Business, Engineering and Science, DCABES 2015, pp. 196–199.

Zhou, A., Yu, D., and Zhang, W. (2015). A research on intelligent fault diagnosis of wind turbines based on ontology and FMECA. Adv. Eng. Inform. 29, 115–125. doi: 10.1016/j.aei.2014.10.001.

Zhou, J., Leppanen, T., Harjula, E., Ylianttila, M., Ojala, T., Yu, C. and Jin, H. (2013) 'CloudThings: A common architecture for integrating the Internet of Things with Cloud Computing', Proceedings of the 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2013, pp. 651–657.

Zhou, X., Tang, X., Yuan, X. and Chen, D. (2009) 'SPBCA: Semantic pattern-based context-aware middleware', Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS, IEEE, pp. 891–895.

ZVEI, (2015) Reference Architecture Model Industrie 4.0 (RAMI4.0):https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/GMA_Status_Report__Reference_Archtitecture_Model_Industrie_4.0__RAMI_4.0_/GMA-Status-Report-RAMI-40-July-2015.pdf.

APPENDICES

Appendix A : Comparison of five major research philosophical stances (Source) adapted from (Saunders et al., 2018)

Philosophical Stance	Philosophical Assumptions			Typical Methods
	Ontology	Epistemology	Axiology	
Positivism	Real, external, independent One true reality Granular Ordered	Scientific method Detectable and quantifiable facts Law-like generalisations Numbers Causal explanation and prediction as contribution	Value-free research Researcher is detached, neutral, and independent of what is researched Researcher maintains objective stance	Typically, deductive, highly structured, large samples Measurement Typically, quantitative methods of analysis, but a range of data can be analysed
Critical Realism	Stratified/layered (the empirical, the actual and the real) External Independent Intransient Objective structures Causal mechanisms	Epistemological relativism Knowledge is historically situated and transient Facts are social constructions Historical causal explanation as contribution	Value-laden research Researcher acknowledges bias by world views, cultural experience and upbringing Researcher tries to minimise bias and errors Researcher is as objective as possible	Reproductive, in-depth historically situated analysis of pre-existing structures and emerging agency Range of methods and data types to fit subject matter
Interpretivism	Complex, rich Socially constructed through culture and language Multiple meanings, interpretations, and realities Flux of processes, experiences, and practices	Theories and concepts too simplistic Focus on narratives, stories, perceptions and interpretations New understandings and worldviews as contribution	Value-bound research Researchers are part of what is researched Subjective Researcher interpretations key to contribution Researcher reflexive	Typically, inductive small samples, in-depth investigations, qualitative methods of analysis, but a range of data can be interpreted
Postmodernism	Nominal Complex, rich Socially constructed through power relations Some meanings, interpretations, and realities are dominated and silenced by others Flux of processes, experiences, and practices	What counts as 'truth' and 'knowledge' is decided by dominant ideologies Focus on absences, silences and oppressed/repressed meanings, interpretations and voices Exposure of power relations and challenge of dominant views as contribution	Value-constituted research Researcher and research embedded in power relations Some research narratives are repressed and silenced at the expense of others Researcher radically reflexive	Typically, deconstructive – reading texts and realities against themselves In-depth investigations of anomalies, silences and absences Range of data types, typically qualitative methods of analysis
Pragmatism	Complex, rich, external 'Reality' is the practical consequences of ideas Flux of processes, experiences and practices	Practical meaning of knowledge in specific contexts 'True' theories and knowledge are those that enable successful action Focus on problems, practices and relevance Problem-solving and informed future practice as contribution	Value-driven research, Research initiated and sustained by researcher's doubts and beliefs, Researcher reflexive	Following research problem and research question Range of methods: mixed, multiple, qualitative, quantitative, action research Emphasis on practical solutions and outcomes

Appendix B (Questionnaire)

EVALUATION OF PREVIOUS KNOWLEDGE

1- To what extent do you know about ontologies? (I.e. what they represent and how they can be used, etc.?)

No knowledge at all	Few notions	Standard Understanding	Good level of knowledge	Expert
☐	☐	☐	☐	☐

2- To what extent do you know about the ontology tool, Protégé, and how to use it?

No knowledge at all	Few notions	Standard Understanding	Good level of knowledge	Expert
☐	☐	☐	☐	☐

3- To what extent do you know about Failure Mode and Effects Analysis (FMEA), (i.e. what they represent and when to use FMEA, etc.?)

No knowledge at all	Few notions	Standard Understanding	Good level of knowledge	Expert
☐	☐	☐	☐	☐

4- To what extent are you familiar with using Semantic Web Rule Language (SWRL) rules?

No knowledge at all	Few notions	Standard Understanding	Good level of knowledge	Expert
☐	☐	☐	☐	☐

5- To what extent have you been directly involved in performing Fault analysis (identifying the failure occurred, components and processes)?

No knowledge at all	Few notions	Standard Understanding	Good level of knowledge	Expert
☐	☐	☐	☐	☐

6- How familiar are you with the terminology used in the fault analysis of mechanical components?

No knowledge at all	Few notions	Standard Understanding	Good level of knowledge	Expert
☐	☐	☐	☐	☐

USABILITY EVALUATION (SUS test)

Please state the degree of your agreement with the following statements.

1- I think that I could contribute to the model presented in this project

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

2- I did not find the model unnecessarily complex

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

3- I find the model easy to understand

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

4- I don't think I would need further theoretical support to be able to understand this model

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

5- I found that various concepts in this system were well-integrated

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

6- I found the model was very consistent

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

7- I would imagine that most Maintenance experts would understand this model very quickly

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

8- I don't find this model very cumbersome to understand

Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

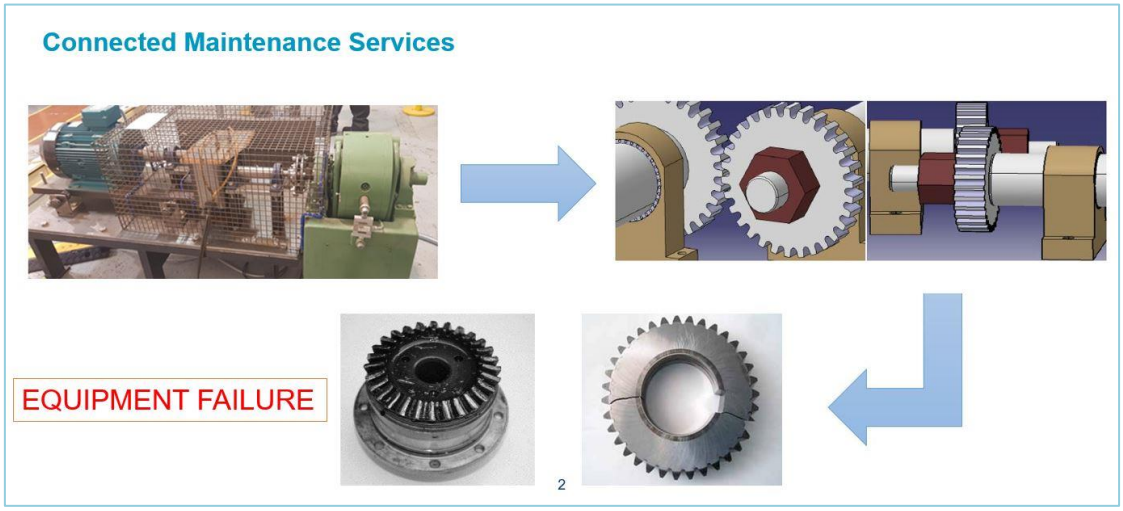
9- I am confident that I understand the conceptualization of this model

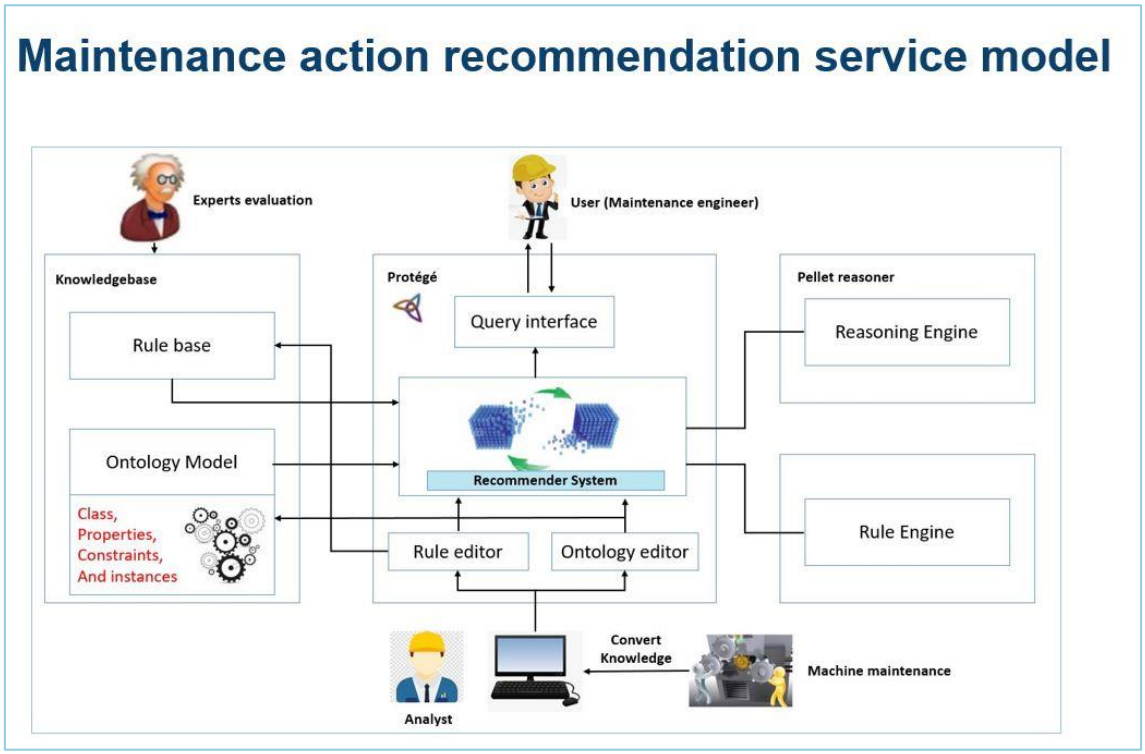
Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

10- I did not need to ask a lot of questions before I could understand the conceptualization of this model.

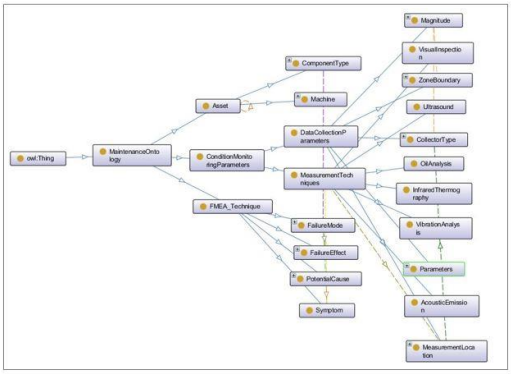
Strongly Disagree	Disagree	Neither Agree nor disagree	Agree	Strongly Agree
☐	☐	☐	☐	☐

Appendix C - Further explanation of the model



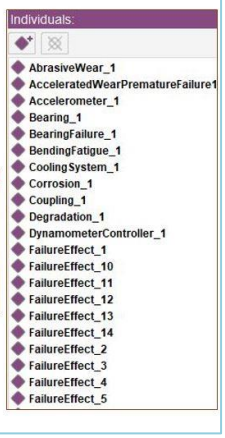


The set of Entities and Properties is implemented on Protégé



Data properties

Individuals



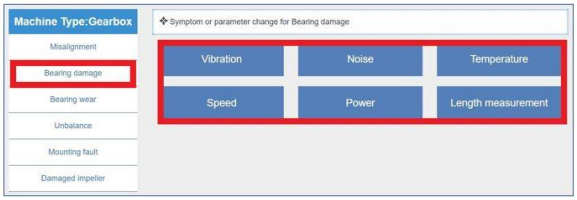
Function, Failure Mode, Failure Cause and Failure Effect for Test Rig.

Shaft

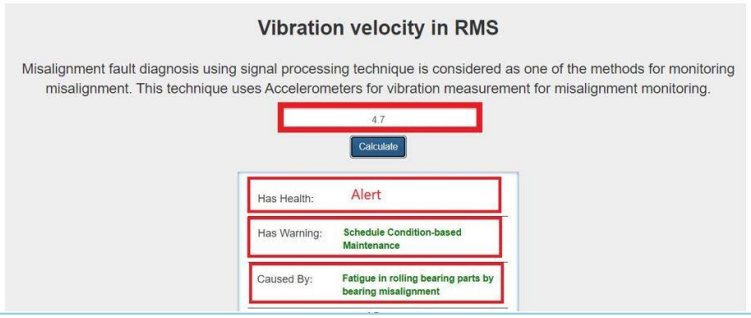
Function | Failure Modes, Effect, and Causes | Mitigation | RPN

(SEV * OCC * DET) = RPN

Failure Mode	Failure Effects	Failure Causes	SEV	OCC	DET	RPN
Bent shaft	Assembly vibration - Damaged bearing, impeller, wear ring, mechanical seal, gear box	Excessive load, torque, Impact loads, Bearing failure	4	5	4	80
Excessive shaft deflection	Eventual shaft damage - Damaged bearing, impeller, wear ring, mechanical seal, gear box	Dynamic loading on shaft - Reversing loads - Critical shaft speed exceeded - Unbalanced load	3	4	3	36
Shaft misalignment	Assembly vibration - Damaged bearing, impeller, wear ring, mechanical seal, gear box	Improper assembly - Worn bearings - Excessive load	3	2	3	18
Damaged surface finish	Shaft bearing failure	Surface cracks, eventual shaft failure - Bearing, gear, coupling corrosion	2	2	2	8
Shaft fatigue / fracture	Stress riser at fillet - Stress concentration at keyway - Shaft radii changes - Bending fatigue - Excessive velocity - High torque load	Damaged bearing, impeller, wear ring, mechanical seal, gear box	5	3	4	60
Fretting corrosion	Relative movement of tightly fitted parts	Surface cracks, eventual shaft failure - Bearing, gear, coupling corrosion	5	3	4	60



Property	SWRL Rules
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Good"^^string) -> has Velocity(?Gp1, "greater than or equal 0 and less than or equal 2.3"^^string)
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Satisfactory"^^string) -> has Velocity (?Gp1, "greater than 2.3 and less than or equal 4.5"^^string)
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Alert"^^string) -> has Velocity (?Gp1, "greater than 4.5 and less than or equal 7.1"^^string)
Has Velocity	Group1 (?Gp1), has Zone (?Gp1, "Alarm"^^string) -> has Velocity (?Gp1, "greater than 7.1"^^string)



Appendix D (DHT22 sensor)

```
import os
import time
import sys
import Adafruit_DHT as dht
import paho.mqtt.client as mqtt
import json
THINGSBOARD_HOST = 'demo.thingsboard.io'
ACCESS_TOKEN = 'IHdVznFPHcrxXz8rA6u6'
# Data capture and upload interval in seconds. Less interval will eventually
hang the DHT22.
INTERVAL=2
sensor_data = {'temperature': 0, 'humidity': 0}
next_reading = time.time()
client = mqtt.Client()
# Set access token
client.username_pw_set(ACCESS_TOKEN)
# Connect to ThingsBoard using default MQTT port and 60 seconds
keepalive interval
client.connect(THINGSBOARD_HOST, 1883, 60)
client.loop_start()
try:
    while True:
        humidity,temperature = dht.read_retry(dht.DHT22, 4)
        humidity = round(humidity, 2)
        temperature = round(temperature, 2)
        print(u"Temperature:                {:g}\u00b0C,                Humidity:
{:g}%".format(temperature, humidity))
        sensor_data['temperature'] = temperature
        sensor_data['humidity'] = humidity
        # Sending humidity and temperature data to ThingsBoard
        client.publish('v1/devices/me/telemetry', json.dumps(sensor_data), 1)
        next_reading += INTERVAL
        sleep_time = next_reading-time.time()
        if sleep_time > 0:
            time.sleep(sleep_time)
except KeyboardInterrupt:
    pass
client.loop_stop()
client.disconnect()
```

Appendix E (Vibration analysis node.js code)

```
var mqtt = require('mqtt');
const csv = require('csv-parser');
var listOfData = [];
var count = 0;
const ACCESS_TOKEN = process.argv[2];
var client  = mqtt.connect('mqtt://demo.thingsboard.io',{
   username: ACCESS_TOKEN
});
var vibrationData = {
   ax: 0,
          ay:0,
          }
client. On('connect', function () {
   console.log('connected');
   console.log ('Uploading data once per second...');
   setInterval(publishTelemetry, 60000);
          getExcelData();
});
function getExcelData(){

          lineReader.eachLine('base.csv', function(row) {
                    console.log(row);
                    var tab = row.split(",");
                              vibrationData.ax = tab[0];
                     vibrationData.ay = tab[1];
                     vibrationData.az = tab[2];
                     vibrationData.aT = tab[3];
                    console.log(JSON.stringify(vibrationData));
                    listOfData.push(JSON.stringify(vibrationData));
          });
          console.log('CSV file successfully processed');
                     setInterval(publishData, 1000);
                      .pipe(csv())
           .on ('line', (row) => {
                     console.log(row);
                    var tab = row.split(",");
                     vibrationData.time = tab[0];
                     vibrationData.ax = tab[1];
                     vibrationData.ay = tab[2];
                     vibrationData.az = tab[3];
                     vibrationData.aT = tab[4];
                    console.log(JSON.stringify(vibrationData));
                    listOfData.push(JSON.stringify(vibrationData));
                    //client.publish('v1/devices/me/telemetry', JSON.stringify(row));
           })
           .on('end', () => {
                    console.log('CSV file successfully processed');
                     setInterval(publishData, 1000);
           }); */
}
```