# Semantic technologies for the modeling of predictive maintenance for a SME network in the framework of industry 4.0

Qiushi Cao

Normandie Université

# THÈSE

**Pour obtenir le diplôme de doctorat**

**Spécialité Informatique**

**Préparée au sein de INSA ROUEN NORMANDIE**

# Semantic Technologies for the Modeling of Predictive Maintenance for a SME Network in the Framework of Industry 4.0


**Présentée et soutenue par**

**Qiushi CAO**


| Thèse soutenue publiquement le 26/06/2020 devant le jury composé de | | |
|---|---|---|
| M. Thomas GUYET | MCF HDR, AGROCAMPUS-OUEST/IRISA | Rapporteur |
| Mme Nada MATTA | PR, Université Technologique de Troyes | Rapporteur |
| Mme Anne HÅKANSSON | PR, UiT - The Arctic University of Norway | Examinateur |
| Mme Samira SI-SAID CHERFI | PR, CNAM | Examinateur |
| Mme Cecilia ZANNI-MERK | PR, INSA Rouen Normandie | Directeur de thèse |
| M. Christoph REICH | PR, Hoschshule Furtwangen | Co-directeur de thèse |
| M. Ahmed SAMET | MCF, INSA Strasbourg | Encadrant de thèse |
| M. François DE BERTRAND DE BEUVRON | MCF, INSA Strasbourg | Encadrant de thèse |

INSA ROUEN NORMANDIE

MIIS

litis

# Acknowledgements

How time flies, that I just noticed, I am at the end of my Ph.D. journey. Three years is almost negligibly short for human history but is considerably long for a human being. Until today, I can still remember those struggles, difficulties, sorrow, and happiness that I have gone through during the pursuit of this Ph.D. degree. Indeed, this dissertation would not be made possible without the help of many people during this precious journey.

Firstly, I would like to give my sincere gratitude to my thesis director, Prof. Cecilia Zanni-Merk. The work with Prof. Cecilia Zanni-Merk is quite enjoyable because she is a helpful and kind-hearted supervisor. Every time I was conflicted with challenging research problems, she was always patient and informative with giving inspiring ideas. Also, she gave me much encouragement and passion when good research results are achieved. These precious experiences and learned lessons will be deeply cherished throughout my future career. I would like to also say thank you to my co-director, Prof. Christoph Reich, for his valuable guidance on my research. It was always entertaining to have a discussion with him on both academia and industrial perspectives. I really learned a lot from him about how to manage an international research project and how to gain industrial-oriented thinking.

Secondly, I would like to thank my thesis co-supervisors, Dr. Ahmed Samet, and Dr. François De Bertrand De Beuvron. Dr. Ahmed Samet opened the domain of machine learning and data mining to me and patiently guided me step by step until I become familiar with these two research fields. He is always helpful with specific technical problems. I learned a lot about how to do research from him. Also, I would like to thank him for his warm host at INSA Strasbourg. Those six months of working with him is a memorable and helpful experience that is beneficial to my research career. I would like to thank Dr. François De Bertrand De Beuvron for his warm welcome in Strasbourg, during my first year in France. He introduced the whole project to me, and gave me fruitful knowledge that is important to my Ph.D. study.

Thirdly, my appreciation goes to the friendly environment and comfortable facilities at the LITIS lab, INSA de Rouen. I would like to thank Ms. Brigitte Diarra, who is always patient and helpful with the administrative affairs. I also want to thank Ms. Delphine Untereiner at INSA Strasbourg, for her management of traveling and reimbursement issues. I understand those documents are quite a lot and are not easy to cope with. A big thanks to all of my colleagues, especially Mr. Mathieu Bourgais, Mr. Henrique Donancio, Mr. Alencar Medeiros Gabriel Henrique, Mr. Usman Malik, Mr. Franco Giustozzi, Mrs. Mirna El Ghosh, Ms. Marwa Kechaou and Mr. Maël Bouabdelli. We had a lot of good times during lunch and daily discussions.

Moreover, my gratitude goes to my best friends, who made this journey far from boring and homesick. My special thanks to FanKa, the Chinese Ph.D. group from where we had endless coffee and endless fun. Credits to my friends Lei Bian, Hao Bai, and Tianxiang Nan for the constructive advice and funny moments we experienced.

Last but not least, I would like to thank my parents, Ms. Yuqin Tang and Mr. Xuesong Cao, for their remote care and support in China. They gave me so much strength and passion for tackling every challenge in my research work and daily life. A big thanks to my girlfriend, Liping Zhao, for her self-giving care and support during my stay in France. Those delicious meals and memorable travels will always be at the bottom of my heart. I wish her Ph.D. study will go smooth as expected. Also, I would like to give my gratitude to my grandparents. I hope they could see my success and be proud of me.

Qiushi Cao
Rouen, France
25/03/2020

# Résumé

Dans le domaine de la fabrication, la détection d'anomalies telles que les défauts et les défaillances mécaniques permet de lancer des tâches de maintenance prédictive, qui visent à prévoir les défauts, les erreurs et les défaillances futurs et à permettre des actions de maintenance. Avec la tendance de l'industrie 4.0, les tâches de maintenance prédictive bénéficient de technologies avancées telles que les systèmes cyberphysiques (CPS), l'Internet des objets (IoT) et l'informatique dématérialisée (cloud computing). Ces technologies avancées permettent la collecte et le traitement de données de capteurs qui contiennent des mesures de signaux physiques de machines, tels que la température, la tension et les vibrations.

Cependant, en raison de la nature hétérogène des données industrielles, les connaissances extraites des données industrielles sont parfois présentées dans une structure complexe. Des méthodes formelles de représentation des connaissances sont donc nécessaires pour faciliter la compréhension et l'exploitation des connaissances. En outre, comme les CPSs sont de plus en plus axées sur la connaissance, une représentation uniforme de la connaissance des ressources physiques et des capacités de raisonnement pour les tâches analytiques est nécessaire pour automatiser les processus de prise de décision dans les CPSs. Ces problèmes constituent des obstacles pour les opérateurs de machines qui doivent effectuer des opérations de maintenance appropriées.

Pour relever les défis susmentionnés, nous proposons dans cette thèse une nouvelle approche sémantique pour faciliter les tâches de maintenance prédictive dans les processus de fabrication. En particulier, nous proposons quatre contributions principales: i) un cadre ontologique à trois niveaux qui est l'élément central d'un système de maintenance prédictive basé sur la connaissance; ii) une nouvelle approche sémantique hybride pour automatiser les tâches de prédiction des pannes de machines, qui est basée sur l'utilisation combinée de chroniques (un type plus descriptif de modèles séquentiels) et de technologies sémantiques; iii) a new approach that uses clustering methods with Semantic Web Rule Language (SWRL) rules to assess failures according to their criticality levels; iv) une nouvelle approche d'affinement de la base de règles qui utilise des mesures de qualité des règles comme références pour affiner une base de règles dans un système de maintenance prédictive basé sur la connaissance. Ces approches ont été validées sur des ensembles de données réelles et synthétiques.

Mots-clés : Industrie 4.0, Maintenance prédictive, Prévision des défaillances de machines, Évaluation de la criticité des défaillances, Ingénierie de l'ontologie, Rule-based Reasoning, Rule Base Refinement, Extraction des chronicles

## Abstract

In the manufacturing domain, the detection of anomalies such as mechanical faults and failures enables the launching of predictive maintenance tasks, which aim to predict future faults, errors, and failures and also enable maintenance actions. With the trend of Industry 4.0, predictive maintenance tasks are benefiting from advanced technologies such as Cyber-Physical Systems (CPS), the Internet of Things (IoT), and Cloud Computing. These advanced technologies enable the collection and processing of sensor data that contain measurements of physical signals of machinery, such as temperature, voltage, and vibration.

However, due to the heterogeneous nature of industrial data, sometimes the knowledge extracted from industrial data is presented in a complex structure. Therefore formal knowledge representation methods are required to facilitate the understanding and exploitation of the knowledge. Furthermore, as the CPSs are becoming more and more knowledge-intensive, uniform knowledge representation of physical resources and reasoning capabilities for analytic tasks are needed to automate the decision-making processes in CPSs. These issues bring obstacles to machine operators to perform appropriate maintenance actions.

To address the aforementioned challenges, in this thesis, we propose a novel semantic approach to facilitate predictive maintenance tasks in manufacturing processes. In particular, we propose four main contributions: i) a three-layered ontological framework that is the core component of a knowledge-based predictive maintenance system; ii) a novel hybrid semantic approach to automate machinery failure prediction tasks, which is based on the combined use of chronicles (a more descriptive type of sequential patterns) and semantic technologies; iii) a new approach that uses clustering methods with Semantic Web Rule Language (SWRL) rules to assess failures according to their criticality levels; iv) a novel rule base refinement approach that uses rule quality measures as references to refine a rule base within a knowledge-based predictive maintenance system. These approaches have been validated on both real-world and synthetic data sets.



Keywords: Industry 4.0, Predictive Maintenance, Machinery Failure Prediction, Failure Criticality Assessment, Ontology Engineering, Rule-based Reasoning, Rule Base Refinement, Chronicle Mining

# Contents

# List of publications

During this Ph.D, the following research papers have been published:

**Peer-reviewed journal papers**

- **Q. Cao**, F. Giustozzi, C. Zanni-Merk, F. De Beuvron, and C. Reich, "Smart Condition Monitoring for Industry 4.0 Manufacturing Processes: an Ontology-based Approach," Cybernetics and Systems: An International Journal, Volume 50, Issue 2: Adding Smartness to Systems With Case Studies and Applications, Online ISSN: 1087-6553, pp.82-96. doi: 10.1080/01969722.2019.1565118

- **Q. Cao**, A. Samet, C. Zanni-Merk, F. De Beuvron, and C. Reich, "Using Rule Quality Measures for Rule Base Refinement in Knowledge-based Predictive Maintenance Systems," Cybernetics and Systems: An International Journal, Volume 51, Issue 2, Online ISSN: 1087-6553, pp.161-176. doi: 10.1080/01969722.2019.1705550

- **Q. Cao**, A. Samet, C. Zanni-Merk, F. De Beuvron, and C. Reich, "Combining Chronicle Mining and Semantics for Predictive Maintenance in Manufacturing Processes," Semantic Web Journal, Special Issue on Semantic Web of Things for Industry 4.0, ISSN: 1570-0844. (Accepted, available online:http://www.semantic-web-journal.net/system/files/swj2440.pdf).

**Peer-reviewed international conference papers**

- **Q. Cao**, C. Zanni-Merk, and C. Reich, "Ontologies for Manufacturing Process Modeling: A Survey," in the 5th International Conference on Sustainable Design and Manufacturing (SDM 2018), Gold Coast, Australia, 24-26 June, 2018. doi: 10.1007/978-3-030-04290-5_7

- **Q. Cao**, C. Zanni-Merk, and C. Reich, "Towards an Ontological Representation of Condition Monitoring Knowledge in the Manufacturing Domain," in the 10th International Conference on Knowledge Engineering and Ontology Development (KEOD 2018), Seville, Spain, 18-20 September, 2018. doi: 10.5220/0006957903100316

- **Q. Cao**, C. Zanni-Merk, and C. Reich, "Towards a Core Ontology for Condition Monitoring," in the 7th International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV 2018), Nancy, France, 8-10 October, 2018. doi: 10.1016/j.promfg.2018.12.029

- **Q. Cao**, "Semantic Technologies for the Modeling of Condition Monitoring Knowledge in the Framework of Industry 4.0," in the doctoral consortium of the 21th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2018), Nancy, France, 12-16 November, 2018

- **Q. Cao**, A. Samet, C. Zanni-Merk, F. De Beuvron, and C. Reich, "An Ontology-based Approach for Failure Classification in Predictive Maintenance Using Fuzzy C-means and SWRL Rules", in the 23rd International Conference on Knowledge-Based and Intelligent Information Engineering Systems (KES 2019), Budapest, Hungary, 4-6 September, 2019. doi: 10.1016/j.procs.2019.09.218

- **Q. Cao**, A. Samet, C. Zanni-Merk, F. De Beuvron, and C. Reich, "Combining Evidential Clustering and Ontology Reasoning for Failure Prediction in Predictive Maintenance," in Proceedings of the 12th International Conference on Agents and Artificial Intelligence (ICAART 2020), Valletta, Malta, 22-24 February, 2020

- G. H. A. Medeiros, **Q. Cao**, C. Zanni-Merk, and A. Samet, "Manufacturing as a Service in Industry 4.0: A Multi-Objective Optimization Approach," in Proceedings of the 12th International Conference on Intelligent Decision Technologies, Split, Croatia, 17-19 June, 2020

# List of Figures

# List of Tables

# Part I

# Synthèse de la thèse en français

# Chapitre 1

# État de l'art

## 1.1 Contexte de la thèse

### 1.1.1 Contexte du projet

Durant la fabrication, la détection précoce d'anomalies permet de lancer des tâches de *maintenance prédictive*, qui visent à prédire les défauts, erreurs et pannes futurs et permettent également de planifier des actions de maintenance. Normalement, une tâche de maintenance prédictive est basée sur la surveillance d'un paramètre de diagnostic mesurable du système, qui identifie l'état d'un système [GDBR02]. De cette façon, des décisions de maintenance, telles que l'appel d'un opérateur, sont proposées en fonction de la gravité des anomalies afin d'éviter les arrêts de la chaîne de production et de minimiser les pertes économiques. Plusieurs techniques ont été utilisées pour détecter l'usure des unités mécaniques et pour prédire les conditions futures des machines, telles que l'apprentissage machine, fouille de données, les statistiques et la théorie de l'information [CBK09].

Toutefois, à mesure que les données recueillies deviennent plus hétérogènes et complexes, il devient difficile pour les opérateurs de réagir aux défaillances mécaniques en temps utile et avec précision. Dans le contexte de l'industrie 4.0, des techniques avancées telles que l'internet industriel des objets (IIoT), les systèmes cyber-physiques (CPS) et l'informatique en nuage (Cloud Computing) permettent d'interconnecter les machines et les systèmes de production dans des usines intelligentes pour échanger des données en continu. Cette tendance a donné aux fabricants la capacité de gérer et d'utiliser efficacement de grandes collections de données. En même temps, cette tendance a induit la demande de méthodologies permettant de

détecter automatiquement les anomalies sur les lignes de production.

Au sein de cette perpective de l'Industry 4.0, cette thèse est réalisée dans le cadre du projet européen Interreg HALFBACK [1]. L'objectif principal du projet HALFBACK est d'assurer la haute disponibilité des processus de fabrication en anticipant les défaillances des machines et des outils, la perte de qualité des produits, les problèmes de flux de ressources, etc. et en programmant la maintenance, le remplacement des composants, la replanification des processus, voire la reprise de la production par une autre usine, de manière optimisée et intelligente. La figure 1.1 montre l'architecture globale du projet.

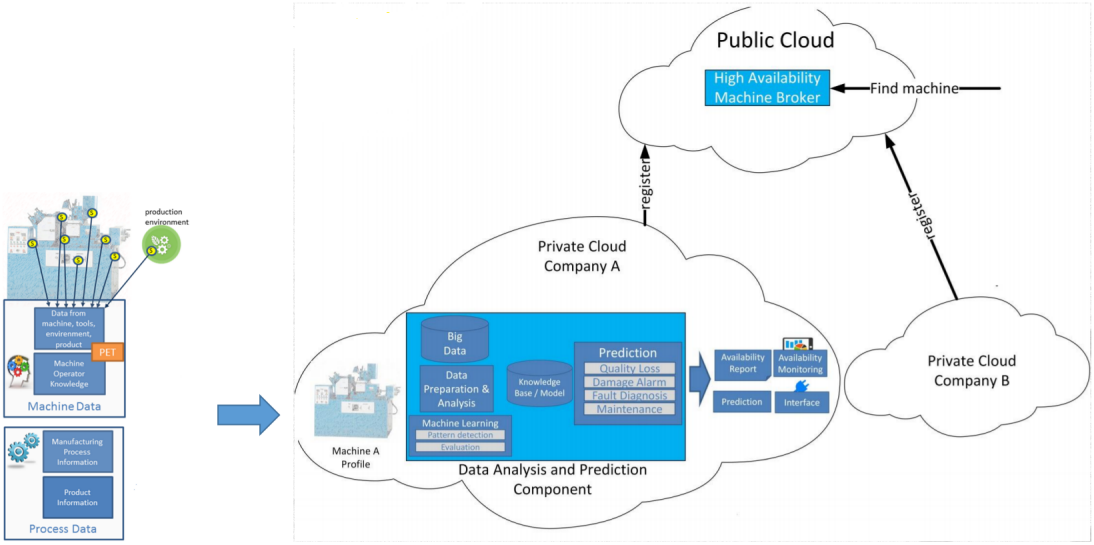



FIGURE 1.1 – L'architecture globale du projet HALFBACK.

Dans le cadre du projet, de grandes quantités de données industrielles sont recueillies à l'aide de capteurs situés sur les machines et les outils. Des informations supplémentaires sont collectées à partir de divers outils, de l'environnement de fabrication, du produit lui-même ainsi que de l'expérience de l'opérateur de la machine. Des algorithmes d'apprentissage adaptés aux mégadonnées analysent les données collectées, et sont associés à des technologies sémantiques pour comprendre le processus en prenant en compte l'expérience des opérateurs, permettant finalement de prédire les dommages aux machines, la perte de qualité ou les demandes de maintenance dans le futur. Cela permet aux entreprises d'agir avant que le processus de fabrication ne s'arrête.

En outre, les profils virtuels des machines (empreintes) sont agrégés dans le

[1] http ://halfback.in.hs-furtwangen.de/home/

nuage et enregistrés auprès d'un "courtier en machines à haute disponibilité". L'enregistrement de l'empreinte de la machine avec, en autre, son emplacement et de sa disponibilité, permet au courtier de proposer la machine comme un service à d'autres entreprises. En cas de panne inévitable d'une machine, le logiciel HALFBACK utilise le "Courtier en machines à haute disponibilité" pour rechercher un remplacement adéquat de la machine afin de transférer la production dans une autre usine pour garantir une haute disponibilité.

Le projet HALFBACK vise à améliorer la compétitivité des petites et moyennes entreprises (PME) manufacturières le long du Rhin par leur mise en réseau avec cette approche innovante de la fabrication en tant que service.

Dans le cadre du projet HALFBACK, cette thèse est réalisée au Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes (LITIS) de l'Institut National des Sciences Appliquées (INSA Rouen)/Normandie Université, France. Les travaux sont menés en collaboration avec le Laboratoire des Sciences de l'Ingénieur, de l'Informatique et de l'Imagerie (ICUBE) à l'INSA de Strasbourg, France, et le laboratoire de recherche Could (Institute for Cloud Computing and IT Security (IFCCITS)) à l'Université de Furtwangen (HFU), Allemagne.

### 1.1.2 Contributions de cette thèse

Pour répondre à ces défis, cette thèse a du aborder des problématiques appartenant à plusieurs domaines de recherche. C'est pourquoi ses principales contributions, esquissées ci-dessous, ont été proposées dans de multiples domaines :

- Développement de l'ontologie : dans cet axe de travail, un cadre ontologique, coeur du système de maintenance prédictive basé sur la connaissance, a été développé. Ce cadre se focalise sur une représentation ontologique des connaissances en matière de maintenance prédictive dans le domaine de la fabrication. Il comprend une ontologie de référence pour représenter les concepts et les relations générales de maintenance prédictive et un ensemble d'ontologies de domaine pour formaliser les connaissances spécifiques au domaine de la fabrication et de la surveillance des conditions.

- Prévision de la défaillance des machines : pour automatiser les tâches de prévision de la défaillance dans l'industrie, une nouvelle approche sémantique hybride basée sur l'utilisation combinée de la fouille de données et des technologies sémantiques a été introduite. Dans le cadre de l'approche séman-

tique, l'extraction de données est utilisée pour prédire les futures défaillances des machines industrielles surveillées, et les ontologies de domaine avec leurs extensions basées sur des règles sont utilisées pour prédire les contraintes temporelles des défaillances et pour représenter les résultats prédictifs de manière formelle.

- Classification des défaillances de machines : en plus de prédire les contraintes temporelles des défaillances de machines, nous nous intéressons également à l'identification de la criticité des défaillances. La classification de la criticité des défaillances permet de lancer des signaux d'alerte à différents niveaux, grâce auxquels les opérateurs de machines peuvent hiérarchiser les actions de maintenance pour les défaillances de niveau de criticité supérieur par rapport à celles de niveau inférieur. Dans cette thèse, une nouvelle approche pour classer les défaillances selon leur niveau de criticité est proposée. L'approche est mixte : basée sur des algorithmes de classification et des technologies sémantiques : les algorthmes de classification sont utilisés pour apprendre la criticité des défaillances en se basant sur les données historiques de la machine, et les technologies sémantiques utilisent les résultats de l'apprentissage de la machine pour prédire le moment des défaillances et leur criticité.

- Affinement de la base de règles : Lorsque le nombre de règles extraites des données historiques des machines est important, cela peut réduire l'efficacité du raisonnement basé sur les règles d'un système basé sur la connaissance. Pour réduire le nombre de règles extraites et pour obtenir une meilleure qualité de la base de règles, une approche basée sur l'optimisation multi-objectifs est proposée. L'approche vise à maximiser les critères *Précision* et *Couverture* pour élaguer une base de règles.

- Capitalisation de l'expérience : les modèles de connaissance dans un système basé sur la connaissance peuvent souffrir de la question de l'incomplétude. Pour surmonter ce problème, une nouvelle approche de la capitalisation de l'expérience qui saisit l'expérience des experts sous forme de règles d'experts a été proposée. Dans le cadre de l'approche proposée, une méthode d'intégration de la base de règles pour combiner les règles de chronique et les règles d'experts est développée. Des critères fondés sur la *Redondance*, la *Contradiction*, et la *Subsomption* sont définis et devront être détectés pour permettre une intégration cohérente de la base de règles.

## 1.2 Maintenance prédictive dans le contexte de l'industrie 4.0

Dans ce chapitre, nous donnons les connaissances de base de la maintenance prédictive pour l'industrie 4.0. Nous présentons d'abord les principaux composants de l'industrie 4.0. Nous démontrons ensuite le concept de systèmes cyberphysiques (CPS) et présentons l'architecture à 5 niveaux d'un CPS. Enfin, nous abordons l'utilisation d'un CPS dans le contexte de la maintenance prédictive de l'industrie 4.0.

Le terme "Industrie 4.0" désigne la tendance actuelle de l'échange et du traitement automatiques des données dans les usines de fabrication. Introduit à l'origine en Allemagne, le terme est rapidement devenu un mot à la mode à l'échelle mondiale [WSJ17]. Suivant la notion de technologie d'automatisation introduite lors de la troisième révolution industrielle, l'industrie 4.0 vise à utiliser les technologies Internet pour créer des produits intelligents, une production intelligente et des services intelligents [WSJ17]. La figure 1.2 montre l'histoire des révolutions industrielles et les techniques clés au sein de chaque révolution. Avec la tendance actuelle de l'industrie 4.0, les CPS, l'internet des objets, le "Cloud Computing" et les techniques d'analyse des "mégadonnées" sont devenus les composants clés qui permettent l'interconnexion automatique et l'échange de données entre les entités manufacturières.

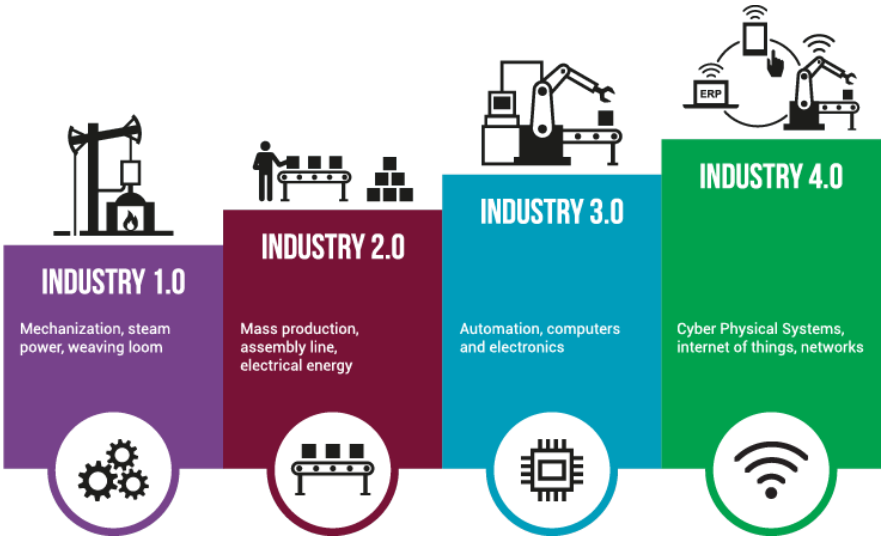



FIGURE 1.2 – Histoire de la révolution industrielle. [2]

[2] https://www.netobjex.com/how-humans-are-empowering-digital-transformation-in-industry-

Dans le secteur manufacturier actuel, la concurrence mondiale croissante, l'évolution rapide des technologies et la perception qu'ont les clients de la qualité des produits nécessitent des plans stratégiques d'évolution pour utiliser des techniques de fabrication avancées. Dans l'industrie 4.0, l'échange et l'analyse automatiques des données ouvrent aux fabricants des possibilités d'optimiser davantage les processus de production. La collecte de données provenant de divers composants d'une chaîne de production et leur analyse dans une infrastructure Cloud évolutive peut améliorer considérablement la productivité, la fiabilité et la disponibilité des systèmes de production dans des environnements hétérogènes [LBK15]. Avec la tendance actuelle à l'automatisation et à l'échange de données dans les technologies de fabrication, les usines de fabrication traditionnelles se transforment en usines dites "intelligentes", qui appliquent des technologies de détection et de calcul de pointe à différents processus de fabrication et systèmes de production.

## 1.3 Modèles existants pour la maintenance prédictive industrielle

Au cours des dernières décennies, un nombre considérable d'efforts de recherche ont été entrepris pour aborder le développement de différents modèles pour la maintenance prédictive industrielle. Pour une tâche de maintenance prédictive, le choix d'un modèle approprié est vital pour les fabricants. Une sélection appropriée d'un modèle nécessite non seulement une compréhension mathématique approfondie de celui-ci, mais aussi la connaissance de la manière de le mettre en œuvre dans des scénarios du monde réel.

Dans ce chapitre, nous classons les modèles existants pour la maintenance prédictive industrielle en quatre catégories : i) les modèles basés sur la connaissance ; ii) les modèles physiques ; iii) les modèles basés sur les données ; et iv) les modèles hybrides. Nous présentons également les avantages et les inconvénients de chaque type de modèles [SHM11].

La Fig. 1.3 résume notre classification des approches de maintenance prédictive les plus courantes, où les approches existantes sont structurées en quatre niveaux, marqués par des couleurs différentes. Le rectangle de couleur bleue représente la super classe abstraite de l'ensemble des méthodes de maintenance prédic-

4-0/

tive.Nous la divisons en quatre sous-classes principales (les rectangles de couleur verte). Ensuite, des modèles plus spécifiques sont présentés par les rectangles de couleur jaune. Enfin, les modèles sont classés en techniques plus spécifiques, qui sont représentées par les rectangles gris.

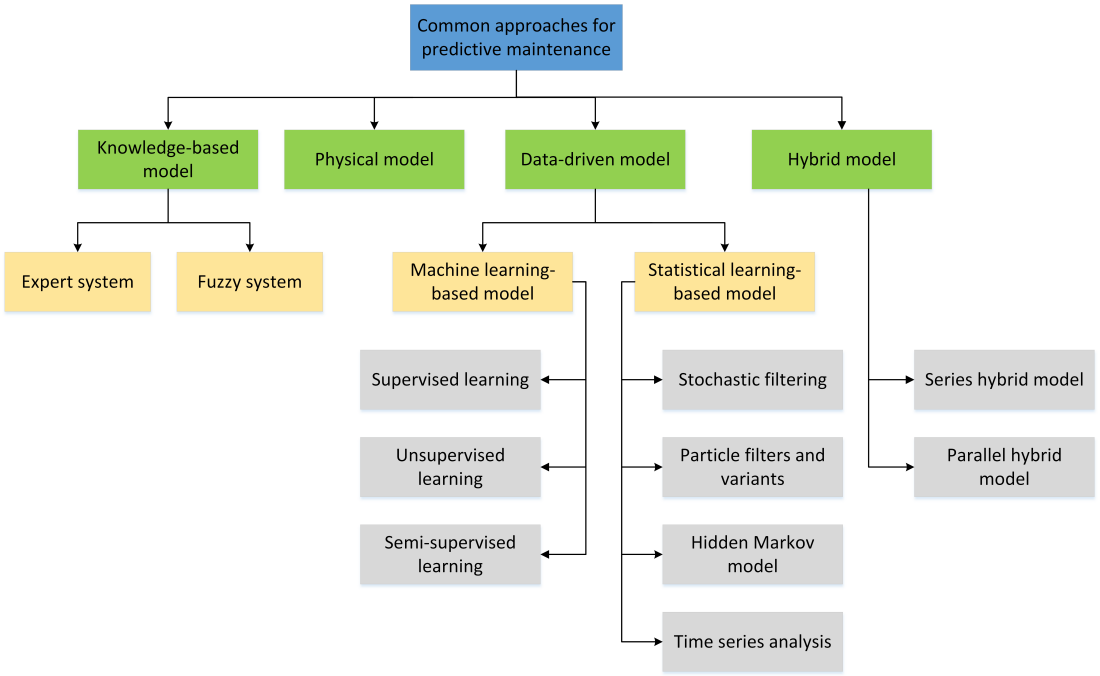



FIGURE 1.3 – Classification des approches communes en maintenance prédictive.

## 1.4 Systèmes fondés sur la connaissance : fondements théoriques

Un système fondé sur la connaissance (KBS *knowledge-based system*) maintient une base de connaissances qui stocke les symboles d'un modèle de calcul sous forme d'énoncés sur le domaine, et effectue un raisonnement en manipulant ces symboles [Gri09]. Les systèmes basés sur la connaissance utilisent des représentations explicites de la connaissance sous forme de mots et de symboles, ce qui facilite la compréhension de la connaissance par un humain ou un ordinateur. De plus, ils aident à résoudre le problème du fossé sémantique (articulation entre les données numériques et les connaissances conceptuelles) et assurent l'interopérabilité sémantique entre les différents systèmes et utilisateurs.

De nos jours, les organisations et les entreprises du secteur manufacturier sont de plus en plus axées sur la collaboration et la connaissance [OBGM11]. Cependant, une grande partie de ces précieuses connaissances sont cachées et ne sont pas directement accessibles aux utilisateurs. Récemment, de nombreuses entreprises et organisations ont réalisé l'importance de la saisie, de la structuration, de la représentation et de la réutilisation des connaissances pour parvenir à des prises de décision intelligentes [Lei10]. Dans ce contexte, les systèmes basés sur la connaissance sont des outils puissants pour résoudre des problèmes complexes du monde réel.

Dans cette thèse, nous visons à développer un système basé sur la connaissance pour faciliter les tâches de maintenance prédictive. Dans ce chapitre, nous présentons les bases théoriques et les concepts de base des systèmes basés sur la connaissance, y compris l'architecture classique et la nouvelle architecture KREM (Knowledge, Rules, Experience, and Meta-Knowledge) [ZM15, ZMS19].

## 1.5 Systèmes de maintenance prédictive basés sur la connaissance : état de l'art

Dans ce chapitre, nous faisons un examen complet des systèmes existants de maintenance prédictive basés sur la connaissance. Nous accordons une attention particulière aux modèles ontologiques et à leurs extensions basées sur des règles qui sont pertinents pour la maintenance prédictive. L'examen des travaux de recherche existants est classé en trois catégories : i) modèles ontologiques et systèmes à base de règles pour la modélisation du domaine de la fabrication ; ii) modèles ontologiques et systèmes à base de règles pour la modélisation du domaine de la maintenance conditionnelle ; iii) modèles ontologiques et systèmes à base de règles pour la modélisation du contexte de fabrication.

L'examen est effectué selon la méthodologie proposée dans [TDS03]. La méthodologie adoptée fournit une approche systématique pour guider l'examen d'un sujet particulier. Cette approche comporte trois étapes : i) la planification de l'examen ; ii) la réalisation de l'examen ; et iii) la communication et la diffusion des résultats [TDS03]. Nous commençons par mener une étude de cadrage pour identifier la pertinence de la littérature dans le domaine de la maintenance prédictive. L'étude de cadrage comprend l'identification des mots clés, la clarification des règles d'inclusion et d'exclusion et la sélection des bases de données. En conséquence, les travaux

connexes qui sont pertinents pour la maintenance prédictive sont collectés et synthétisés.

Après l'examen approfondi de la littérature, nous présentons dans les sous-sections suivantes l'état de l'art des modèles ontologiques et de leurs extensions basées sur des règles qui sont liées à la maintenance prédictive. Nous précisons la portée de ces systèmes en identifiant le domaine (fabrication, surveillance des conditions ou contexte) qu'ils servent.

Nous résumons la couverture du domaine par les modèles de connaissance existants (ontologies et leurs extensions basées sur des règles) dans le tableau 1.1. Nous évaluons la couverture du domaine et la portée de ces modèles en examinant si les concepts clés nécessaires pour décrire le domaine de la maintenance prédictive sont couverts et décrits formellement. Si un concept est couvert par un modèle de connaissance, une coche est placée dans le tableau. Dans le cas contraire, une croix est apposée.

Après avoir examiné les modèles de connaissances mentionnés ci-dessus, nous voyons qu'aucun d'entre eux ne fournit une représentation satisfaisante des connaissances des trois sous-domaines. Certains de ces modèles de connaissance se concentrent sur un domaine étroit, tel que la planification des ressources de fabrication, et ils ne formalisent pas les concepts liés à la maintenance prédictive, par exemple, les états *Failure* et *Fault*. En outre, aucun des modèles de connaissance existants ne fournit une représentation des concepts liés à *Signal d'alerte* dans les tâches de maintenance, par exemple, *Alerte* et *Alarme*. Pour effectuer une tâche de maintenance prédictive sur une machine, la base de connaissances d'un système basé sur la connaissance devrait incorporer non seulement les connaissances interprétables par la machine pour caractériser les entités ou les processus de fabrication qui sont surveillés, mais aussi les connaissances sur la détection et le pronostic des défauts ou des défaillances. Cela nous motive à développer un modèle de connaissance plus expressif et plus complet qui fournit une représentation riche des connaissances du domaine dans les domaines de la fabrication, de la surveillance des conditions et du contexte.

TABLEAU 1.1 – Un résumé des modèles de connaissance existants en ce qui concerne leur couverture du domaine.

| Ontologies | Context | | | | Condition Monitoring | | | | | | | | Manufacturing | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Identity | Activity | Time | Location | Anomaly | Fault | Failure | Severity | Prognostics | Diagnostics | Alarm | Alert | Product | Process | Resource |
| MASON [LSD+] | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MSDL [AD08] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MRO [UYC+13] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| ONTO-PDM [VHL05] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MCCO [UYC+11] | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ |
| MaRCO [JSHL19] | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MSE [LH07] | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ |
| ADACOR [BL07] | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| Wind turbine ontology [ZYZ15] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Sensing System Ontology [MBB+18] | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| OntoProg Ontology [NB17] | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ |

# Chapitre 2

# Contributions

## 2.1 Un cadre ontologique pour les systèmes de maintenance prédictive basés sur la connaissance

Dans ce chapitre, un cadre pour le développement d'un système de maintenance prédictive fondé sur la connaissance est présenté. Ce cadre est centré autour d'une représentation ontologique des connaissances en matière de maintenance conditionnelle dans le domaine de la fabrication. Cette représentation comprend une ontologie de référence pour les concepts et relations générales de la maintenance conditionnelle, ainsi qu'un ensemble d'ontologies de domaine pour formaliser les connaissances plus spécifiques liées en particulier au contexte de la fabrication. L'ontologie de référence centrale est alignée sur l'ontologie UFO (Unified Foundational Ontology), qui est une ontologie de niveau supérieur fournissant des concepts généraux et des relations à un niveau d'abstraction élevé [GW10]. Les ontologies de domaine spécialisent l'ontologie de référence dans les domaines de la fabrication et de la surveillance des conditions, avec la représentation des connaissances spécifiques aux aspects tels que le contexte, le produit, le processus ou les ressources.

La figure 2.1 montre l'ensemble du cadre ontologique. Les ontologies y sont structurées dans l'architecture à trois niveaux représentée par les rectangles avec des lignes pointillées, et correspondant, du bas vers le haut, à des degrés d'abstraction croissants. Les rectangles arrondis avec des lignes pleines sont des ontologies ou des modèles conceptuels différents. Les flèches pleines indiquent l'alignement entre les ontologies.

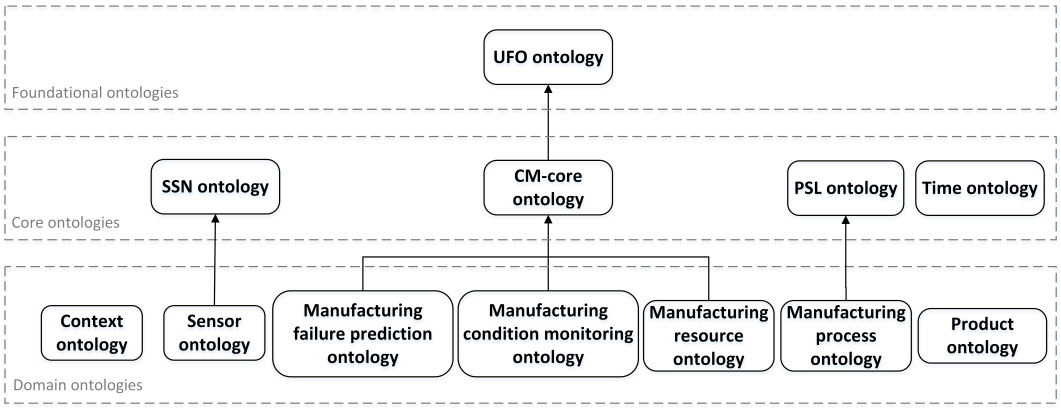



FIGURE 2.1 – Le cadre ontologique à trois niveaux.

Le cadre ontologique est élaboré selon une approche “Middle-Out” [RPKC11]. Il comprend une ontologie qui modélise les concepts et les propriétés de base pour la surveillance des conditions, appelée CM-core. L'ontologie CM-core contient les concepts de base liés à la surveillance des conditions, tels que le *Système*, la *Structure*, le *Comportement*, la *Fonction*, le *Défaut* et la *Défaillance*. La généralité de l'ontologie CM-core assure la possibilité de sa spécialisation dans des ontologies de domaine plus spécifiques, telles que les ontologies pour la surveillance des conditions des machines dans l'industrie. Ainsi, l'ontologie CM-core peut être étendue et personnalisée pour mieux satisfaire les besoins dans des domaines individuels où les ontologies doivent être utilisées pour renforcer les systèmes d'information. L'ontologie CM-core est alignée avec l'ontologie fondamentale UFO, pour obtenir une conceptualisation rigoureuse. De plus, l'ontologie CM-core est spécialisée dans les ontologies de domaine, pour intégrer et représenter les connaissances spécifiques à un domaine.

## 2.2 Une nouvelle approche sémantique hybride pour la maintenance prédictive

Comme le domaine de la maintenance prédictive devient de plus en plus axé sur la connaissance, les tâches effectuées dans ce domaine peuvent souvent bénéficier de l'incorporation de la connaissance du domaine et du contexte, grâce à laquelle la sémantique des résultats de la recherche de chroniques par fouille de donnée peut être explicitement représentée et clairement interprétée. Les chro-

niques sont un type de modèles séquentiels, où les événements sont ordonnés, et les contraintes temporelles parmi ces événements sont décrites comme des intervalles de temps [SMS+19]. Toutefois, à notre connaissance, aucun travail n'a été proposé pour combiner l'extraction de chroniques et la sémantique afin de faciliter la maintenance prédictive des processus de fabrication. De plus, la plupart des travaux de recherche existants sur la maintenance prédictive dans le domaine de la fabrication se concentrent simplement sur la classification des conditions de fonctionnement des machines (par exemple, condition de fonctionnement normal, condition de panne...), tout en manquant d'extraction d'informations temporelles spécifiques sur l'occurrence des pannes [SMS+19]. Cela crée des obstacles pour les utilisateurs qui doivent effectuer des actions de maintenance en tenant compte des contraintes temporelles.

Ce chapitre propose une nouvelle approche sémantique hybride. L'approche sémantique utilise des ontologies (présentées au chapitre 2) avec leurs extensions basées sur des règles pour représenter les résultats de l'extraction des chroniques dans un format sémantique riche, qui améliore la représentation et la réutilisation des connaissances. En spécifiant la sémantique du domaine et en annotant les données industrielles avec une sémantique riche et formelle, les ontologies avec leurs extensions basées sur des règles aident à résoudre les problèmes décrits ci-dessus. Plus précisément, ce chapitre décrit les contributions suivantes :

- Une nouvelle approche sémantique hybride pour automatiser les tâches de prédiction des pannes de machines, qui est basée sur l'utilisation combinée de chroniques et de technologies sémantiques.

- Un nouvel algorithme pour transformer les chroniques en règles logiques basées sur le SWRL, par lequel les résultats prédictifs sont formalisés, donc interprétables à la fois par l'homme et les machines. La transformation proposée permet la génération automatique de règles SWRL à partir des résultats de la recherche des chroniques par fouille de données, permettant ainsi une approche sémantique automatique pour la prédiction des défaillances des machines.

- Une évaluation de la faisabilité et de l'efficacité de l'approche sémantique proposée en menant des expériences sur un ensemble de données industrielles réelles. La performance de la construction des règles SWRL et la qualité de la prédiction des défaillances sont évaluées par rapport à l'ensemble

des données industrielles.

Le lancement d'un tel ensemble de règles prédictives basées sur le SWRL permet de prévoir les contraintes temporelles des futures défaillances des machines. Cela permet aux utilisateurs de prendre d'autres mesures de maintenance, comme le remplacement des machines-outils utilisées sur la chaîne de production. La performance de la prédiction des défaillances pourrait être améliorée en envisageant un nouvel ensemble de règles qui raisonnent sur les niveaux de criticité des défaillances. Dans le chapitre suivant, des techniques d'apprentissage automatique sont utilisées pour classer les niveaux de criticité des défaillances, en fonction des contraintes temporelles parmi les défaillances et autres événements.

## 2.3 Évaluation des défaillances de machines sur la base des niveaux de criticité

Dans le domaine de la maintenance prédictive, la prédiction et l'évaluation de la criticité des défaillances est une question cruciale pour les fabricants. En obtenant les niveaux de criticité des différentes défaillances, les opérateurs de machines peuvent hiérarchiser les actions de maintenance pour les défaillances de niveau de criticité plus élevé par rapport à celles de niveau inférieur. De cette façon, les résultats de la prédiction des défaillances peuvent être utilisés pour planifier la maintenance des machines. Cependant, les approches de maintenance prédictive existantes dans le domaine de la fabrication se limitent au déploiement de systèmes de surveillance de l'état des machines pour détecter les anomalies et prévoir le moment des futures défaillances des machines, tout en ne disposant pas des solutions pour identifier la criticité des défaillances des machines [AGN19]. Il en résulte un lien manquant entre les informations temporelles d'une anomalie (par exemple, le moment d'une future panne de machine) et la criticité de l'anomalie. Pour évaluer la disponibilité des systèmes de fabrication, la durée de l'arrêt est une considération clé qui indique la criticité d'une défaillance mécanique [ALRL04]. Par conséquent, la prévision des moments de défaillance des machines est cruciale pour calculer la durée de l'arrêt et la criticité des défaillances.

Dans ce chapitre, les travaux du chapitre 2.2 sont prolongés par l'introduction de deux approches hybrides basées sur l'ontologie pour les tâches de prédiction et de classification des défaillances. Les approches hybrides basées sur l'ontologie

sont basées sur l'utilisation combinée de techniques de regroupement et de raisonnement ontologique. Dans le cadre des approches proposées, nous abordons l'incertitude de la criticité des défaillances en adoptant deux cadres d'incertitude : le regroupement flou de *c*-means (FCM) [BEF84] et le regroupement probabiliste de *c*-means (ECM) [MD08b]. Ces outils sont utilisés pour classer les défaillances en fonction de leur niveau de criticité.

Les apports de ce chapitre résident tout d'abord dans la formalisation des connaissances de maintenance prédictive basée sur des ontologies, par lesquelles les résultats des ECM sont formalisés et la criticité des défaillances est déduite. Deuxièmement, la classification des défaillances est réalisée par la mise en œuvre d'une approche d'apprentissage non supervisée. L'approche d'apprentissage non supervisé utilise des algorithmes FCM et ECM pour regrouper les défaillances en fonction de leur moment d'occurrence, reflétant leur criticité. Ensuite, chaque groupe est étiqueté avec un indice de criticité (élevé, moyen ou faible) pour classer les défaillances en fonction de leurs contraintes de temps et du coût de maintenance estimé. Troisièmement, des ontologies et des règles SWRL sont proposées pour formaliser les résultats de la classification, afin de faciliter la représentation et l'interprétation des connaissances pour la maintenance prédictive.

L'approche a été validée sur un ensemble de données industrielles réelles et plusieurs ensembles de données synthétiques.

## 2.4 Utiliser les mesures de qualité des règles pour l'élagage et l'intégration des règles

Comme nous l'avons présenté dans les chapitres 2.2 et 2.3, l'extraction de chroniques fréquentes est une technique prometteuse pour prédire non seulement l'ordre des événements non défaillants mais aussi les intervalles temporels entre eux. Le résultat de l'extraction de chroniques fréquentes est un ensemble de chroniques de défaillances qui se présentent sous la forme de règles logiques. Ce type de règles (dans ce chapitre, elles sont définies comme *règles de la chronique*) décrit différents événements ainsi que leurs contraintes temporelles et prédit le temps d'apparition des futures défaillances.

Normalement, le nombre de règles de chroniques extraites de l'extraction des chroniques fréquentes est important. En raison d'un certain degré d'impréci-

sion dans les données du monde réel, certaines des règles de chronique extraites peuvent souffrir d'une mauvaise qualité. En conséquence, les règles de chronique de faible qualité (par exemple, les règles avec une faible précision dans la prédiction des échecs) peuvent réduire l'efficacité et la précision du raisonnement basé sur des règles. Il est donc nécessaire d'utiliser une méthode d'élagage basée sur les règles pour obtenir un sous-ensemble de règles de haute qualité.

D'autre part, comme les mégadonnées industrielles sont collectées à partir d'une variété d'appareils et d'environnements, les règles de décision qui en sont extraites peuvent être obtenues à partir de sources hétérogènes. Cela peut causer un ensemble de problèmes (par exemple, contradiction de règles, subsomption et redondance) lorsque des règles hétérogènes sont combinées pour obtenir un niveau satisfaisant de performance de raisonnement. L'utilisation d'une base de règles de faible qualité peut entraîner des décisions de maintenance inappropriées, ce qui affaiblit les performances des systèmes de maintenance basés sur la connaissance.

Dans ce chapitre, une nouvelle approche d'affinement de la base de règles est proposée. L'approche proposée consiste en deux méthodes d'affinement des règles qui visent à améliorer progressivement la qualité d'une base de règles :

- Une méthode d'élagage et de réduction des règles appliquée à la base de règles obtenue à partir de l'extraction des chroniques fréquentes. Pour réduire le nombre de règles extraites et obtenir une meilleure qualité de la base de règles, une approche d'optimisation multi-objectifs est appliquée. L'approche vise à maximiser la précision et la couverture des règles pour obtenir un ensemble de règles qualité maximale.

- Une méthode d'intégration des règles pour combiner les règles de chronique et les règles d'expert. Pour améliorer la performance de la prédiction des défaillances, l'expérience des experts doit être capitalisée sous forme de règles d'experts lorsque les règles de chronique ne permettent pas de prendre des décisions correctes. Dans ce travail, nous considérons des règles d'experts qui ont une structure similaire à celle des chroniques. Lorsque ces règles d'experts sont intégrées aux règles de chronique, elles peuvent souffrir de problèmes liés à la redondance, à la contradiction, ou à la subsomption. Dans ce contexte, une méthode d'intégration des règles permettant de détecter ces trois problèmes est proposée.

Les deux méthodes proposées ont été validées sur un ensemble de données industrielles réelles.

## 2.5 Le prototype du software : KSPMI

Pour réaliser et automatiser le pipeline de maintenance prédictive susmentionné, nous avons développé un prototype de logiciel appelé Knowledge-based System for Predictive Maintenance in Industry 4.0 (KSPMI). Le logiciel utilise à la fois des approches inductives (exploitation minière chronique et apprentissage machine) et des approches déductives (ontologies de domaine et raisonnement ontologique) pour analyser les données industrielles et prévoir les défaillances futures. Dans ce chapitre, nous présentons d'abord l'environnement et les outils de développement du prototype du logiciel. Ensuite, nous introduisons les fonctionnalités de base de celui-ci en suivant les étapes clés de la chaîne de maintenance prédictive : extraction de chroniques, transformation/génération de règles SWRL, élagage de règles SWRL, capitalisation de l'expérience/intégration de règles d'experts et prédiction des défaillances.

## 2.6 Conclusions et travaux futurs

### 2.6.1 Contributions de ce travail de thèse

Avec la vision "industrie 4.0", l'industrie manufacturière bénéficie aujourd'hui d'une tendance à l'automatisation des échanges de données. Les CPS sont au centre de cette vision et au coeur des usines intelligentes, où les installations de production sont capables d'échanger des informations de manière autonome et intelligente. Dans les usines intelligentes, les machines de production sont connectées pour construire des CPS, qui constituent une nouvelle catégorie de systèmes techniques offrant une interaction étroite entre les composants cybernétiques et physiques. L'échange et l'analyse automatiques des données offrent aux fabricants la possibilité d'optimiser davantage les processus de production. La collecte de données provenant des différents composants d'une chaîne de production et leur analyse dans une infrastructure Cloud évolutive peut améliorer considérablement la productivité, la fiabilité et la disponibilité des systèmes de production dans des environnements hétérogènes. Toutefois, l'utilisation de ces technologies avancées

offre non seulement les avantages susmentionnés aux fabricants, mais leur pose également des défis, tels que la gestion de données volumineuses et hétérogènes générées par des machines et des capteurs en réseau.

Cette vision a été réalisée dans le cadre du projet européen Interreg HALFBACK [1], qui vise à assurer des processus de fabrication hautement disponibles, en prévoyant les pannes de machines, d'outils, la perte de qualité des produits, les problèmes de flux de ressources, etc. et en programmant la maintenance, le remplacement des composants, la replanification des processus, voire la reprise de la production par une autre usine, de manière optimisée et intelligente.

À mesure que les CPS deviennent de plus en plus complexes, les connaissances requises pour l'exploitation et la maintenance des systèmes deviennent elles aussi de plus en plus complexes. Dans ce contexte, des modèles standard et bien définis pour la saisie de ces connaissances complexes sont requis. Pour développer un tel modèle, la connaissance du domaine de la fabrication et de la maintenance prédictive doit être structurée de manière formelle, rendant ainsi cette connaissance utilisable par un CPS. En outre, le domaine de la fabrication étant de plus en plus axé sur les connaissances, une représentation uniforme des ressources physiques et des capacités de raisonnement est nécessaire pour automatiser les processus décisionnels dans les CPS. Ces processus décisionnels comprennent l'intégration automatique des ressources, la prédiction et le diagnostic des anomalies, la programmation de la maintenance et la replanification des processus. Pour réaliser cette vision, les technologies sémantiques ont montré des résultats prometteurs en formalisant les connaissances sur les tâches de maintenance prédictive dans divers domaines.

La nouvelle approche sémantique proposée dans cette thèse permet la représentation des résultats de la fouille de données dans un format formel et structuré, facilitant ainsi la compréhension et l'exploitation des connaissances extraites. De cette façon, les résultats des données peuvent être interprétés à la fois par les utilisateurs et les machines pour enrichir et améliorer les bases de connaissances dans les systèmes de maintenance prédictive basés sur la connaissance.

### 2.6.2 Perspectives

Au delà des contributions résumées dans la section précédente cette thèse ouvre des perspectives de recherches futures, en particulier :

[1] http ://halfback.in.hs-furtwangen.de/home/

- Le premier travail futur est l'évolution de l'ontologie et de la base de règles. Le domaine de la fabrication étant très dynamique, un système de maintenance prédictive devrait être capable de s'adapter à des situations dynamiques dans le temps, par exemple, le changement de contexte. Pour cela, l'ontologie et la base de règles doivent être capables de faire face à l'évolution dynamique des connaissances. Pour traiter cette question, des solutions d'évolution de la base de connaissances vont être proposées : l'ontologie développée dans le chapitre 2 devrait être capable de s'adapter efficacement aux changements grâce à l'utilisation de techniques d'évolution de l'ontologie [SMMS02], et la base de règles extraite de l'extraction de chroniques (introduite dans le chapitre 2.2) devrait être mise à jour en fonction du changement de contexte, en mettant en œuvre un raisonnement contextuel [BBH$^+$10].

- Le deuxième travail futur est la prise en compte d'un plus grand nombre de mesures de la qualité des règles. Au chapitre 2.3, une approche d'optimisation multi-objectifs qui vise à sélectionner des règles avec un maximum de *précision* et de *couverture* a été proposée. À l'avenir, nous souhaitons impliquer davantage de mesures de qualité des règles pour l'élagage des règles, telles que *Association*, *Information*, et *Suffisance logique* [AC01].

- La troisième perspective d'avenir est l'évaluation de la qualité des règles pour les règles d'experts. En effet, en raison des caractéristiques dynamiques et incertaines du domaine de la fabrication, les experts peuvent fournir des règles d'experts erronées pour la prédiction des défaillances. Les règles d'experts erronées peuvent conduire à des résultats de prédiction de défaillance incorrects. Pour résoudre ce problème, un processus d'évaluation est nécessaire pour examiner la performance des règles d'expertise. De cette façon, un ensemble de règles de meilleure qualité est sélectionné non pas à partir de la base de règles des chroniques, mais à partir de la base de règles intégrées, ce qui assure une meilleure performance dans la prédiction.

- La quatrième perspective est la capacité du système à traiter des données en temps réel. Comme le domaine de la fabrication est très dynamique, la possibilité de traiter des flux de données hétérogènes en temps réel est une préoccupation cruciale pour les fabricants. Cependant, l'approche proposée utilise les techniques classiques de raisonnement ontologique, qui ne permettent pas de traiter des données hautement dynamiques en temps voulu.

Pour faire face à ce problème, des techniques de raisonnement en flux devraient être adoptées pour raisonner sur une variété de données hautement dynamiques [DDVvHB17]. Dans le raisonnement par flux, des langages d'interrogation riches sont fournis par les raisonneurs de flux pour interroger en continu les flux de données. De cette façon, les systèmes de maintenance prédictive pourront détecter et prévoir les pannes de machines en temps réel.

# Part II

# Introduction and state of the art

# Chapter 3

# Introduction

## Contents

This chapter gives a general introduction to this thesis. It starts with background information for this thesis, followed by the motivation for conducting this research work. After that, the main contributions of this thesis are presented. Finally, the thesis structure is presented in detail.

## 3.1 Background

Manufacturing processes are sets of structured operations to transform raw material or semi-finished product parts into further completed products. To ensure high productivity, availability, and efficiency of manufacturing processes, the detection of harmful tendencies and conditions of production lines is a crucial issue for manufacturers. In general, anomaly detection on production lines is performed by analyzing data collected by sensors, which are located on machine components and also in production environments. The collected data record real-time situations and reflect the correctness of mechanical system conditions. When the tendency of a mechanical failure emerges, experienced operators in factories are able to take appropriate operations to prevent the outage situations of production systems.

However, as the collected data become more heterogeneous and complex, it is conceivable that the machine operators may fail to respond to mechanical failures timely and accurately. In the context of Industry 4.0, advanced techniques such as the Industry Internet of Things (IIoT), Cyber-Physical Systems (CPS), and Cloud Computing enable machines and production systems in smart factories to be interconnected to exchange data continuously. This trend has brought opportunities for manufacturers to manage and use the collected big data effectively. Meanwhile, this trend has triggered the demand for methodologies to automatically detect anomalies on production lines.

In the manufacturing domain, the detection of anomalies such as mechanical faults and failures enables the launching of *predictive maintenance* tasks, which aim to predict future faults, errors, and failures and also enable maintenance actions. Normally, a predictive maintenance task relies on the monitoring of a measurable system diagnostic parameter, which identifies the state of a system [GDBR02]. In this way, maintenance decisions, such as calling the intervention of a machine operator, are proposed based on the severity of anomalies, to prevent the halt of the production lines, and to minimize economic loss. Several techniques have been used to detect wear and tear in mechanical units and to predict future machinery

conditions, such as machine learning, data mining, statistics, and information theory [CBK09].

With the vision of Industry 4.0, this thesis is carried out under the framework of the European Interreg HALFBACK project [1]. The main goal of the HALFBACK project is to assure highly available manufacturing processes, by forecasting failures of machines, tools, product quality loss, resource flow problems, etc. and by scheduling maintenance, component replacing, process re-planning, and even take over the production by another factory, in an optimized and intelligent way. Fig. 3.1 shows the global architecture of the project.

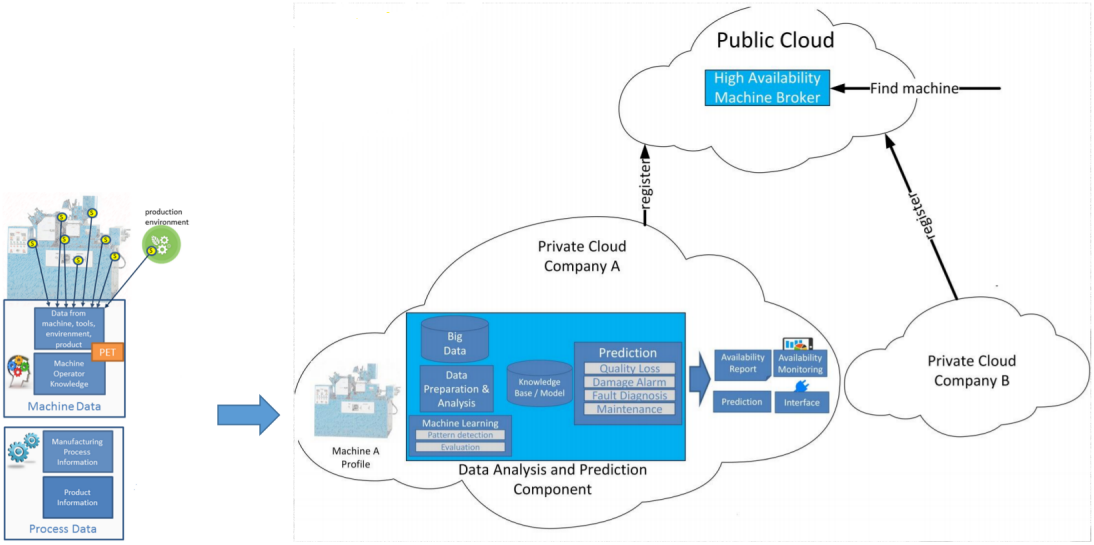



Figure 3.1 – The global architecture of the HALFBACK project.

Within the project, big industrial data is gathered using sensors located on the machines and tools. Additional information is collected from various tools, the manufacturing environment, the product itself as well as the machine operator's experience. Big data algorithms use the collected data to understand the process and to learn from the experience of the operators, and semantic technologies are used to predict machine damage, quality loss, or maintenance demands in the future. This allows companies to act before the manufacturing process stops. Additionally, virtual profiles of the machines (footprints) are aggregated in the cloud and registered at a "High Availability Machine Broker". Registering the machine footprint, the machine location, the machine availability, among other useful data to the broker, allows it to offer the machine as a service to other companies. In case of unavoidable

[1] http://halfback.in.hs-furtwangen.de/home/

machine failure, the HALFBACK software uses the "High Availability Machine Broker" to search for an adequate machine replacement to shift production to another factory in order to guarantee high availability.

The HALFBACK project aims to improve the competitiveness of manufacturing small and medium-sized enterprises (SMEs) along the river Rhine by their networking with this innovative approach to manufacturing as a service.

Within the framework of the HALFBACK project, this thesis is realized at the Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes (LITIS) at Institut National des Sciences Appliquées (INSA Rouen)/Normandie Université, France. The work is conducted with the collaboration of the Laboratoire des Sciences de L'ingénieur, de L'informatique et de L'imagerie (ICUBE) at INSA Strasbourg, France, and the Could Research Lab (Institute for Cloud Computing and IT Security (IFCCITS)) at Furtwangen University (HFU), Germany. Considering the goals of the HALFBACK project, the responsibilities of the teams at different labs are described as follows:

- The team at IFCCITS lab is responsible for the development of the cloud data analysis platform. The platform consists of software agents for collecting data, a CPS gateway module that communicates with a cloud service, a preprocessor module for data preparation, and a data analysis module.

- The team at LITIS lab focuses on using knowledge engineering techniques, especially conceptual representation and inference processes, to develop knowledge-based predictive maintenance systems. This type of systems uses formal conceptual models such as ontologies with a set of logical rules to facilitate predictive maintenance tasks in manufacturing processes.

- The team at ICUBE lab works on the development of data mining algorithms for machinery failure prediction. The data mining algorithms aim to preprocess, mine, and use frequent valuable patterns to predict the failures of machines at appropriate time instants.

In this thesis, we focus on the second axis of work, which concerns the use of knowledge engineering techniques for developing knowledge-based predictive maintenance systems.

## 3.2 Motivation

With the trend of Industry 4.0, predictive maintenance tasks are benefiting from a cyber-physical approach. Within a cyber-physical system (CPS), production facilities are able to exchange information with autonomy and intelligence, which enable manufacturers to optimize the production processes. Fig. 3.2 shows the architecture of a CPS designed for predictive maintenance tasks. Within a CPS, predictive maintenance of manufacturing entities is performed based on a three-layer collaboration between the cyber space and the physical space:

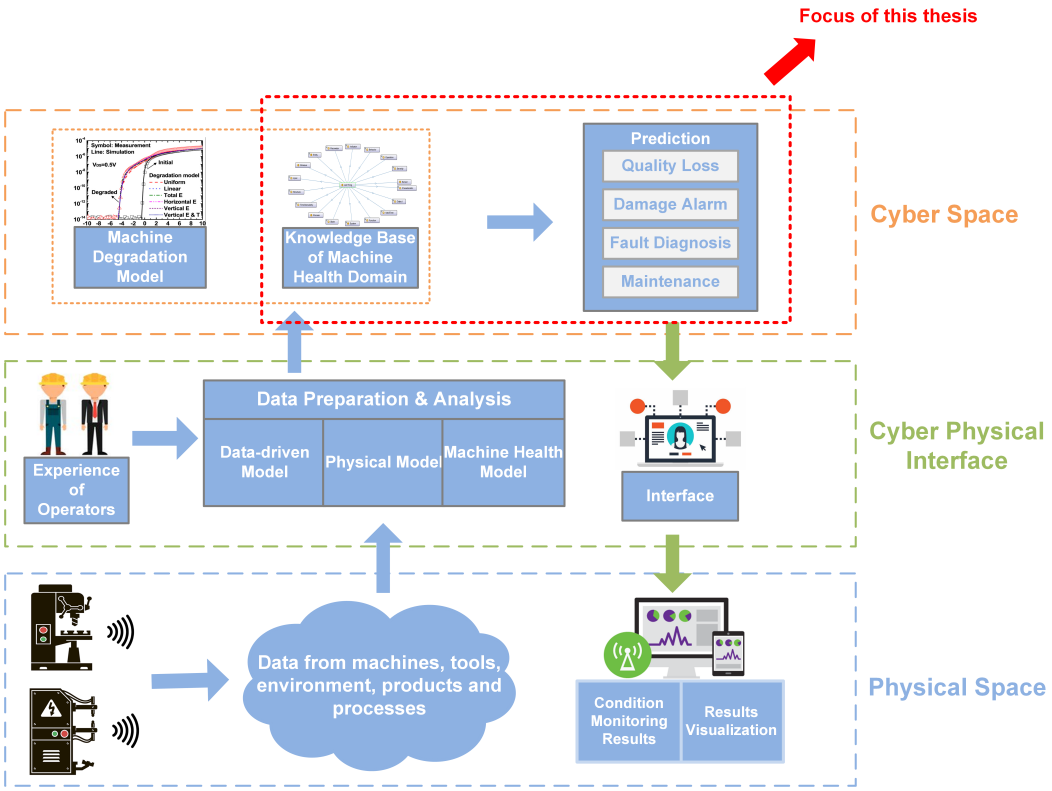



Figure 3.2 – The layered cyber-physical approach for predictive maintenance in the HALFBACK project.

- The Physical Space, where machine operating data is gathered using sensors located on the machines and machine components. Additional data is collected from the products, manufacturing environments, as well as the machine operators' experience.

- The Cyber-Physical Interface, where statistical techniques such as data min-

ing and machine learning use the collected data to understand the manufacturing processes and to learn from operators' experience.

- The Cyber Space, where decision-making about machine failure prediction and maintenance are proposed. In this layer, machine degradation models, and knowledge base of machine health are employed to predict machine damage, quality loss or maintenance demands in the future.

In the second layer of the architecture, data mining is normally performed by collecting and processing sensor data that contain measurements of physical signals of machinery, such as temperature, voltage, and vibration. By identifying events and patterns that are not consistent with the expected behavior, potential hazards in production systems, such as power outage of the systems, could be detected [GDBR02].

However, due to the 3Vs (volume, variety, and velocity) traits of big industrial data, sometimes the knowledge extracted from data mining is presented in a complex structure. Therefore formal knowledge representation methods are required to facilitate the understanding and exploitation of the knowledge [RP16]. Furthermore, as the CPSs are becoming more and more knowledge-intensive, uniform knowledge representation of physical resources and reasoning capabilities are needed to automate the decision-making processes in CPSs [WXY+17]. These issues bring obstacles to machine operators to perform appropriate maintenance actions. To overcome these issues, semantic technologies have been used in several research efforts to promote the interpretation and management of knowledge [DWL15, RP16, GDBR02, CGZM+19, DZWL16]. In their works, semantic technologies devoted to enabling the encoding of semantics with the data in order to make the data both machine and human-readable. For example, in [DZWL16], semantic technologies are used to establish an ontology-based framework for construction risk knowledge management in a Building Information Modelling (BIM) environment. The framework facilitates knowledge reuse during the construction risk analysis process.

Also, since semantic technologies ensure the explicit representations of machine-interpretable domain semantics, they can support semantic interoperability in a large heterogeneous environment of loosely coupled systems [Obr03]. In the data mining domain, several stages can benefit from the involvement of formal semantics, such as data transformation, algorithm selection, and post-processing

[DWL15]. Moreover, the use of semantic technologies allows the capitalization of experience of domain experts [ZM15]. For example, in a predictive maintenance task of machine cutting tool, when data mining algorithms fail to identify the occurrence time of a future cutter failure, logic-based expert rules which capitalize experience of domain experts can be applied to propose predictive decisions.

In the context of predictive maintenance in smart factories, pattern mining has been widely used to discover frequently occurring temporally-constrained patterns, through which warning signals can be sent to humans for a timely intervention [DD99]. Among pattern mining techniques, chronicle mining has been applied to industrial data sets for extracting temporal information of events [CMM12]. The extracted temporal information is valuable for predicting potential machinery failures that may appear in the future [CMM12]. However, even though chronicle mining results are expressive and interpretable representations of complex temporal information, domain knowledge is required for users to have a comprehensive understanding of the mined chronicles [PHW02]. As the predictive maintenance domain is becoming more knowledge-intensive, tasks performed in this domain can often benefit from incorporating domain and contextual knowledge, by which the semantics of the chronicle mining results can be explicitly represented and clearly interpreted. However, most of the existing research works about predictive maintenance in the manufacturing domain merely focus on the classification of operating conditions of machines (e.g., normal operating condition, breakdown condition), while lacking the extraction of specific temporal information of failure occurrence. This brings obstacles for users to perform maintenance actions with the consideration of temporal constraints.

The aforementioned challenges motivate us to propose a novel semantic approach to facilitate predictive maintenance tasks in manufacturing processes (shown as "Focus of this thesis" in Fig. 3.2). The proposed approach aims to combine data mining and semantic technologies for automating the predictive maintenance tasks. Within the approach, data mining technologies such as chronicle mining is used to discover valuable patterns from industrial data. On the other hand, semantic technologies, especially domain ontologies with their rule-based extensions, are used to predict the temporal constraints of failures and to represent the predictive results formally.

## 3.3 Contributions of this thesis

Based on the motivation, in this thesis, the following contributions have been proposed in multiple research fields:

- Ontology development: in this axis of work, an ontological framework that is the core component of a knowledge-based predictive maintenance system has been developed. The framework is developed based on an ontological representation of predictive maintenance knowledge in the manufacturing domain. It includes a core reference ontology for representing general predictive maintenance concepts and relations and a set of domain ontologies for formalizing domain-specific knowledge of manufacturing and condition monitoring.

- Machinery failure prediction: to automate the failure prediction tasks in industry, a novel hybrid semantic approach that is based on combined use of data mining and semantic technologies has been introduced. Within the semantic approach, chronicle mining is used to predict the future failures of the monitored industrial machinery, and domain ontologies with their rule-based extensions are used to predict temporal constraints of failures and to represent the predictive results formally.

- Assessment of failure criticality: in addition to predicting the temporal constraints of machinery failures, we are also interested in identifying the criticality of the failures. The assessment of failure criticality enables the launching of warning signals with different levels, by which machine operators can prioritize maintenance actions for higher-criticality-level failures compared to lower-level ones. In this thesis, a novel approach to classify failures according to their criticality levels is proposed. The approach is based on clustering and semantic technologies, within which clustering techniques are used to learn the criticality of the failures based on machine historical data, and semantic technologies use the machine learning results to predict the time of failures and the criticality of them.

- Rule base refinement: When the number of rules extracted from machine historical data is large, it may reduce the efficiency of rule-based reasoning of a knowledge-based system. To reduce the number of extracted rules and to

achieve the best quality of a rule base, a multi-objective optimization-based approach is proposed. The approach aims to maximize rule *Accuracy* and *Coverage* to prune a rule base, for obtaining a set of rules with the best quality.

- Experience capitalization: the knowledge models in a knowledge-based system may suffer from the incompleteness issue. To overcome this problem, a novel experience capitalization approach that captures the experts' experience in the form of expert rules has been proposed. Within the proposed approach, a rule base integration method for combining chronicle rules and expert rules is developed. Rule *Redundancy*, *Contradiction*, and *Subsumption* are considered as the crucial issues that needs to be detected and eliminated during the rule base integration process.

## 3.4 Structure of the thesis

We use Fig. 3.3 to show the structure of this thesis. Organized into three main parts, this thesis consists of ten chapters:

- **Part I: Synthèse de la thèse en français** gives the summary of this thesis in French.
  - **Chapter 1** gives the context and state of the art of this thesis in French.
  - **Chapter 2** introduces the contributions of this thesis in French.
- **Part II: Introduction and state of the art** introduces the theoretical foundations and gives a comprehensive review on existing research works. It is structured into four chapters.
  - **Chapter 3** gives a general introduction to background, motivation, research scope, and main contributions of this thesis.
  - **Chapter 4** gives the background knowledge of Industry 4.0 predictive maintenance. We first introduce the key components of Industry 4.0, including CPS, the Internet of Things (IoT), Cloud Computing and Big Data analysis techniques. We then pay special attention to CPS, which is the central technology for Industry 4.0 predictive maintenance.

  - **Chapter 5** introduces the existing approaches for Industry 4.0 predictive maintenance. We classify the existing models for industrial predictive maintenance into four categories: i) knowledge-based models; ii) physical models; iii) data-driven models; and iv) hybrid models. We also present the advantages and disadvantages of each type of models, as well as typical applications.
  - **Chapter 6** gives the theoretical foundations and basic concepts of knowledge-based systems. We mainly introduce the KREM architecture, which is a novel and generic knowledge-based framework for problem-solving in engineering disciplines.
  - **Chapter 7** presents a comprehensive review of the existing knowledge-based predictive maintenance systems. In the review, we pay special attention to the ontological models and their rule-based extensions that are relevant to predictive maintenance.

- **Part III: Contributions** demonstrates the contributions of this thesis. It consists of five chapters.

  - **Chapter 8** introduces the ontological framework that is developed for the knowledge-based predictive maintenance system. The framework includes a core reference ontology for representing general predictive maintenance concepts and relations and a set of domain ontologies for formalizing domain-specific knowledge of manufacturing and condition monitoring.
  - **Chapter 9** introduces the novel hybrid semantic approach for machinery failure prediction. Within the approach, chronicle mining is used to predict the future failures of the monitored industrial machinery, and domain ontologies with their rule-based extensions are used to predict temporal constraints of failures and to represent the predictive results formally. A case study on a real-world industrial data set shows the approach in detail.
  - **Chapter 10** demonstrates the failure criticality assessment approach. We use machine learning techniques such as fuzzy c-means and evidential clustering tools to identify failure criticality according to different parameters. The approach has been validated on one real-world industrial

data set and several synthetic data sets.

- **Chapter 11** corresponds to a novel rule base refinement approach. It consists of a multi-objective optimization-based approach for rule base pruning/reduction, and an rule integration approach for combining chronicle rules with expert rules. For rule pruning, rule *Accuracy* and *Coverage* are considered as reference measures for obtaining a set of rules with the best quality. On the other hand, the rule integration approach refines the integrated rule base by detecting issues such as rule *Redundancy*, *Contradiction*, and *Subsumption*. It ensures the rule base is progressively updated and refined to achieve better performance for failure prediction. The proposed approach is validated on a real-world industrial data set, with several simulated expert rules as input.
- **Chapter 12** introduces a software prototype we have developed to automate and facilitate predictive maintenance in industry 4.0. It is named as Knowledge-based System for Predictive Maintenance in Industry 4.0 (KSPMI). The software uses both inductive approaches (chronicle mining and machine learning) and deductive approaches (domain ontologies and ontology reasoning) to analyze industrial data and to predict future failures.
- **Chapter 13** concludes the thesis and outlines future perspectives.

- **Part IV: Appendix** gives the theory of three algorithms/tools that are used in this thesis: the fuzzy $c$-means clustering algorithm, the theory of evidential clustering, and the fast non-dominated sorting algorithm.

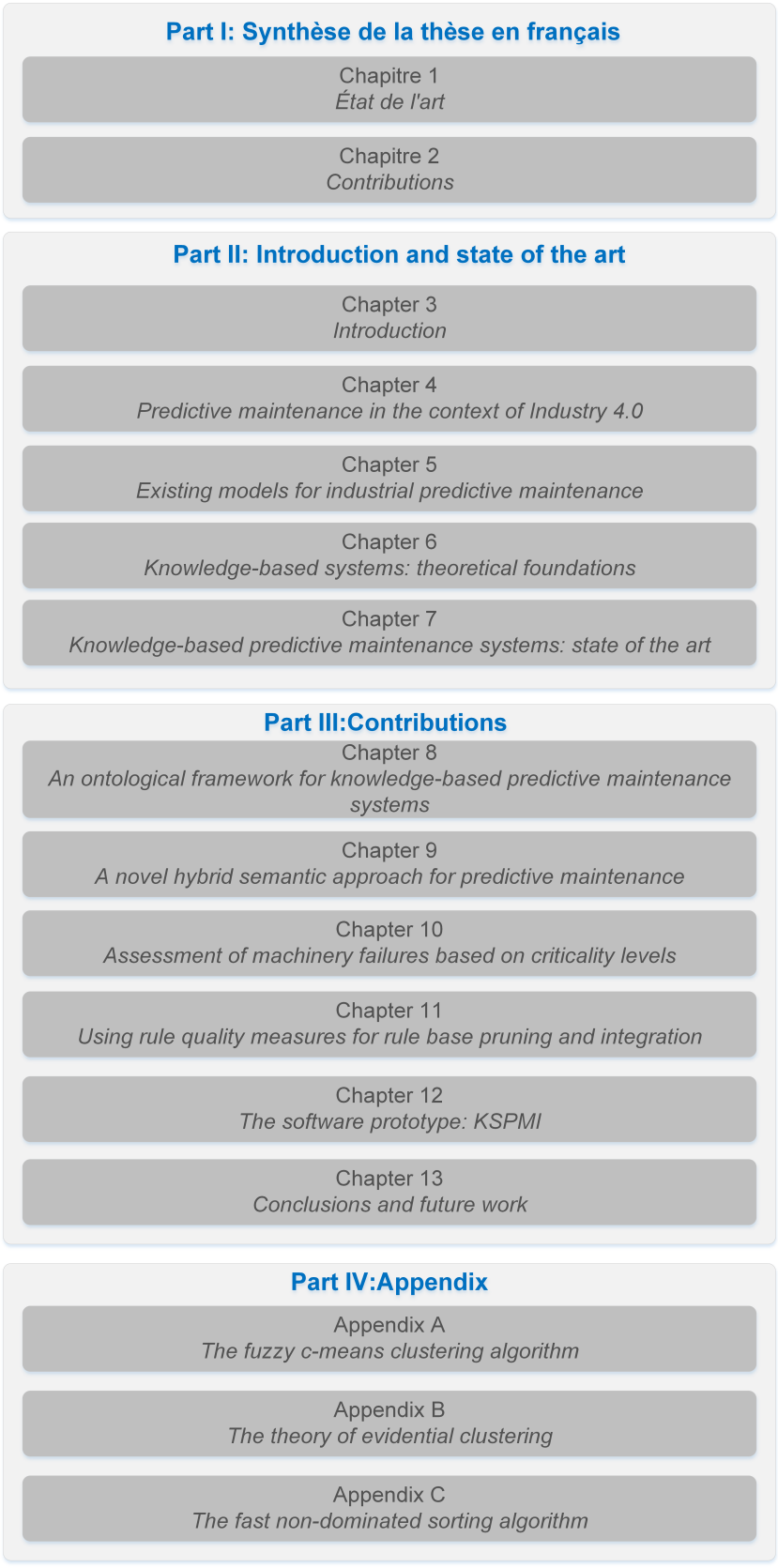



Figure 3.3 – Structure of the thesis.

# Chapter 4

# Predictive maintenance in the context of Industry 4.0

## Contents

## 4.1 Introduction

In this chapter, we give the background knowledge of Industry 4.0 predictive maintenance. We first introduce the key components of Industry 4.0. We then demonstrate the concept of Cyber-Physical Systems (CPS) and present the 5-level architecture of a CPS. At last, we discuss the use of a CPS in the context of industry 4.0 predictive maintenance.

## 4.2 Key components of Industry 4.0

The term "Industry 4.0" stands for the current trend of automatic data exchange and processing in manufacturing factories. Originally introduced in Germany, the term quickly became a buzzword on a global scale [WSJ17]. Following the notion of automation technology that is introduced in the third industrial revolution, Industry 4.0 aims to use Internet technologies to create smart products, smart production, and smart services [WSJ17]. Fig. 4.1 shows the history of industrial revolutions and the key techniques within each revolution. With the current trend of Industry 4.0, CPS, the Internet of Things (IoT), Cloud Computing and Big Data analysis techniques have become the key components that allow the automatic interconnection and data exchange among manufacturing entities.

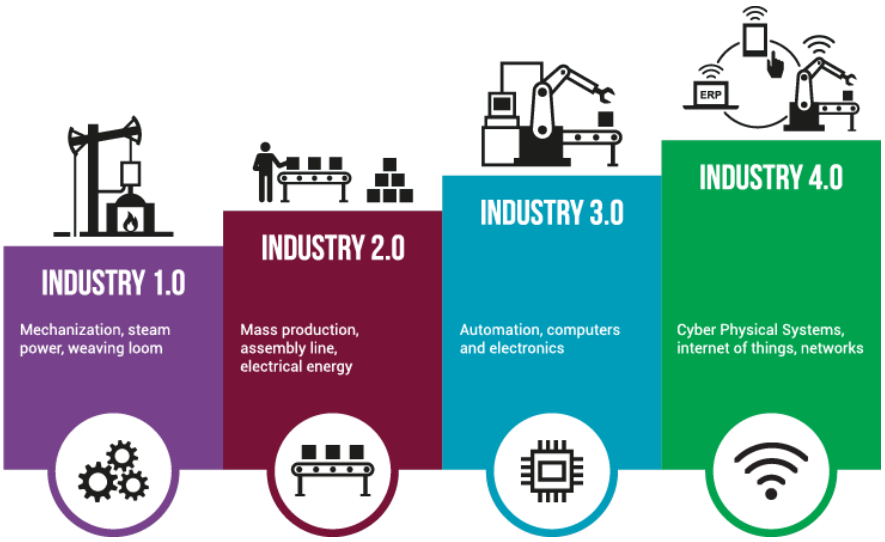



Figure 4.1 – History of industrial revolution. [1]

[1] https://www.netobjex.com/how-humans-are-empowering-digital-transformation-in-industry-4-0/

In today's manufacturing, increasing global competition, fast technology evolution and customers' perceptions of product quality have triggered the demand for future strategical plans and advanced manufacturing techniques. In Industry 4.0, the automatic exchange and analysis of data open up opportunities for manufacturers to further optimize the production processes. Collecting data from various components of a production line and analyzing them in a scalable Cloud infrastructure can significantly improve the productivity, reliability, and availability of production systems in heterogeneous environments [LBK15]. With the current trend of automation and data exchange in manufacturing technologies, traditional manufacturing factories are transforming into so-called "smart factories", which apply high-tech sensing and computation technologies on different manufacturing processes and production systems.

### 4.2.1 The Internet of Things (IoT)

In general, IoT refers to the networked interconnection of everyday objects, which are often equipped with ubiquitous intelligence [XYWV12]. An IoT system consists of a highly distributed network of devices, in which every device can communicate and interact with human beings and other devices via embedded systems. By this, IoT increases the ubiquity of the Internet [XYWV12]. It is estimated that by 2020, the Internet of Things (IoT) will contribute $1.9 trillion to the global economy [McC15].

To address the demand for advanced digitalization within factories, the combined use of Internet technologies and future-oriented technologies results in a new fundamental paradigm shift in industrial production [LFK+14]. This has brought the trend of using the Industrial Internet of Things (IIoT) technologies, within which the data exchange and processing is undergoing a tremendous change. In IIoT, industrial objects are equipped with electronics and smart devices. Technologies such as Radio Frequency Identification (RFID) and smart wearables provide identification, smart computing, and communication capabilities for industrial objects. By this, IIoT allows stakeholders to use everyday Internet-enabled devices as endpoints for accessing industrial data, thus enabling real-time monitoring and maintenance of the physical assets in smart factories [WSJ17]. Moreover, the IIoT provides a higher level of organization and management of industrial value chains and enables highly flexible and resource-saving production services [SWW15].

To illustrate the use of IIoT in industry, we use Fig. 4.2 to show the architecture of

an IIoT-based production system in a smart factory. Within the production system, products are equipped with smart devices, and information related to the product (e.g., identity, specification, process control, product optimization) is stored in some backend-database [SWW15]. These smart devices are capable of storing and processing the data that are generated not only during production but also during the product deployment phase. In this way, the production system becomes self-organizing, that they can optimize themselves with regard to resource planning, availability, and consumption, even across multiple manufacturing companies.

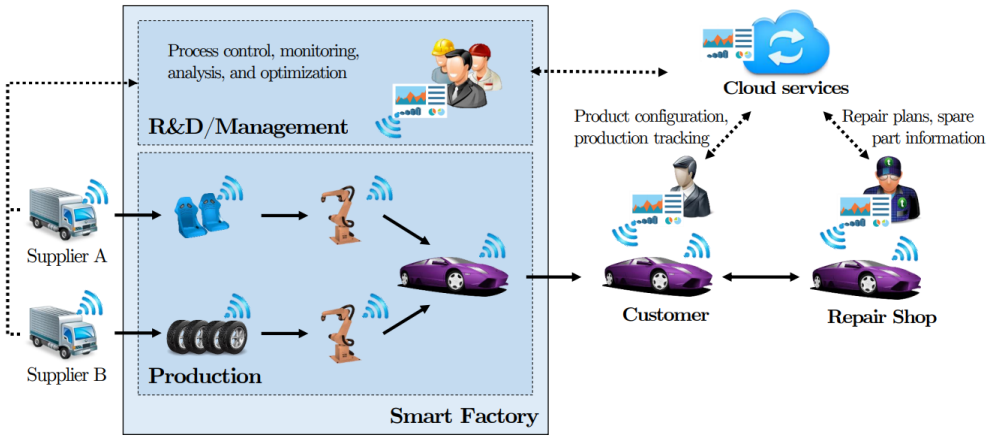



Figure 4.2 – The architecture of an IIoT-based production system [SWW15].

By using wireless sensor devices to monitor equipment and product status, IIoT systems can be pervasively used to collect, store and process data, thus improving the productivity, reliability, and safety of production systems through advanced automatic processes [CBS+17]. Moreover, the use of smart wireless devices avoids the delay caused by human-in-the-loop interactions. This is accomplished by adopting several communication standards that ensures the devices to interoperate. The main communication protocols and standards include [DXHL14]: RFID (e.g., ISO 18000 6c EPC class 1 Gen2) [PLLL+08], IEEE 802.11 (WLAN) [PC03], IEEE 802.15.4 (ZigBee) [MBC+04], IEEE 802.15.1 (Bluetooth) [XP09], IETF Low power Wireless Personal Area Networks (6LoWPAN) [SB11], and traditional IP technologies such as IPv6 [Hui98], etc.

### 4.2.2 Cyber-Physical Systems (CPS)

As another key component of Industry 4.0, Cyber-Physical Systems (CPS) have been widely adopted in a variety of production processes. This type of systems tightly in-

tegrate components in the physical space (physical assets, sensors, actuators) with advanced software and algorithms in the cyber space, thus transferring the physical world into the virtual one. Within a typical CPS, industrial data are collected by the sensors located on entities in the physical space, while computational resources are used to integrate and process the collected data. After that, data analysis and processing results are used to provide feedback and decision makings. These results are conveyed to the physical space to guide the real-world production processes.

In general, a CPS consists of two main functional components [LBK15]: i) the advanced connectivity functionality that allows real-time data collection, processing, and communication between the physical space and the cyber space; ii) intelligent data management, analytics, and computational techniques that are capable of providing decision makings in the cyber space. Normally, there are five layers that communicate and interact with each other within a CPS: i) the Smart Connection Layer, where data acquisition procedure is performed to collect accurate and reliable data from machines; ii) the Data-to-Information Conversion Layer, where useful information is inferred from collected data; iii) the Cyber Layer, which acts as a central information hub to gather information. Thereafter, specific analytics is used to extract additional information and provide better insight over the status of physical assets in the physical space; iv) the Cognition Layer, where a thorough knowledge of the monitored system is generated; v) the Configuration Layer, which provides feedback from cyber space to physical space. Based on the feedback, appropriate actions are triggered in the physical space to respond to the feedback.

In Section 4.3, we give a more detailed description of the 5C architecture of a CPS, with introducing applications and techniques associated with different layers.

### 4.2.3 Cloud Computing

Cloud Computing refers to both the applications delivered as services over the Internet and the hardware in the data centers that are used to provide those services [JKK+10]. The services are referred to as Software as a Service (SaaS) or IaaS (Infrastructure as a Service). Within this vision, a cloud performs as a data center. When a cloud is not made available to the general public, it is referred to as a private cloud. On the other hand, when a cloud is made available in a pay-as-you-go manner to the general public, it is called a public cloud [JKK+10].

Within the current trend of Cloud Computing, computational resources are mi-

grating from local PCs to distant data centers reached through the Internet. On the Internet, a client computer can communicate with other servers efficiently at the same time [JKK+10]. In industry, Cloud Computing not only provides scalable computing capacity, but also enables the provision of services that are accessible globally via the Internet [SMH+15]. Because of these advantages, Cloud Computing has become the basis for creating new business processes and models. Moreover, product data that are integrated in the cloud can be used for predictive maintenance and product optimization.

### 4.2.4 Big Data analysis technologies

As today's global market is becoming more and more competitive, manufacturing companies are forced to come up with solutions for smart and rapid decision making. The utilization of advanced technologies such as CPS and Cloud Computing not only offer the aforementioned benefits to manufactures but also brings them challenges such as the management of big and heterogeneous data generated by networked machines and ubiquitous sensors. Data management and analysis in a Big Data environment is of vital importance for the self-awareness and self-learning of machines. In this context, Big Data analysis technologies have been used within a variety of manufacturing processes.

At the early stage of Big Data, plenty of research efforts were addressed in the domain of social or commercial data mining. These works include sale and customer behavior prediction, opinion mining, recommendation systems, etc. [MBD+12, ANAH08, CSB06]. However, these research works focus on the analysis of human-generated data. In industry, there are not only human-related data but also the Big Data generated from machines and manufacturing environments. These data are collected from smart sensors that are located on machine components, machine tools, shop floors, etc. These sensors not only collect data but also are equipped with advanced techniques for processing the collected data. They measure the real-time system diagnostic parameters such as temperature, vibration, and pressure, etc. Also, machine historical data are collected to learn about the past behavior of machines.

To deal with the challenge of industrial Big Data, manufacturers and researchers have devoted themselves to come up with advanced data management and analysis solutions. In terms of data management, Big Data technologies are using new pro-

cessing models to extract valuable information from heterogeneous data sources (e.g., product data, operational data, environment data). These technologies aim to achieve in-depth understanding and gain insight from data for accurate and timely decision making [ZLZ15]. Also, jointly used with Cloud Computing and CPS, Big Data technologies allow production systems to be self-aware and self-maintained. This ensures production systems to self-access their health conditions and the level of degradation, for them to perform smart maintenance decisions to avoid potential hazards and anomalies.

On the other hand, customers' personalized data are collected from the Web in real-time. These data are also uploaded to a cloud data center for analysis. The analysis of customers' data helps to guide and optimize production processes. In this way, Big Data analysis technologies help manufacturers to improve the efficiency of production, reduce the cost, and optimize the manufacturing processes [ZLZ15].

## 4.3 The 5-level architecture of a CPS

As introduced in Section 4.2.2, a CPS consists of two main functional components and is structured into a 5-level architecture. In this subsection, we introduce the 5-level architecture in detail and present the technologies that are used within each level.

We use Fig. 4.3 to demonstrate the functionalities and technologies associated with each level of the CPS architecture. As introduced before, a CPS is structured into five layers:

- The Smart Connection Layer. This layer is responsible for data collection and communication tasks. Industrial big data are collected from sensors, Enterprise Resource Planning (ERP), and Manufacturing Execution Systems (MES). After that, communication protocols and standards are used to transfer data to central servers in the cloud.

- The Data-to-Information Conversion Layer. At this layer, advanced technologies and tools such as data mining and machine learning algorithms are used to extracted useful information from the collected data. Recently, special attention has been paid to the development of prognostics and machine health management applications [LBK15]. One notable implementation is the estimation of the remaining useful life (RUL) of machines [SWHZ11].

- The Cyber Layer. At this layer, information of connected machines is integrated to form a machine network. Within the network, self-comparison capabilities are provided to estimate the condition of a machine by comparing it with those of the same type. Also, similarities between machine real-time data and machine historical data are measured to predict the future behavior of them.

- The Cognition Layer. Knowledge is generated at this layer for a thorough understanding of a monitored system. Proper presentation of the acquired knowledge is needed for the system to propose appropriate decision makings [LBK15].

- The Configuration Layer. This layer aims to provide feedback from the cyber space and convey it to the physical space. The feedback includes predictive and diagnostic decisions over machines. These decisions are proposed at the Cognition Layer.

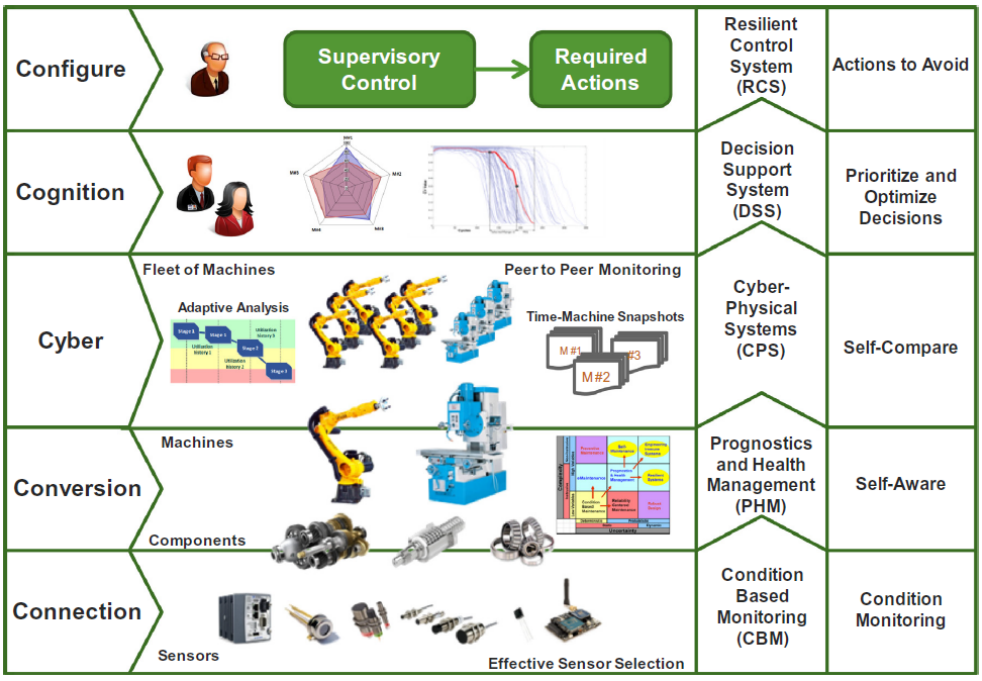



Figure 4.3 – The functionalities and technologies associated with each level of the 5C CPS architecture [LBK15].

## 4.4 Cyber-physical-based predictive maintenance

Manufacturing processes are sets of structured operations to transform raw material or semi-finished product parts into further completed products. To ensure high productivity, availability and efficiency of manufacturing processes, the detection of harmful tendencies and conditions on production lines is of vital importance. In general, anomaly detection on production lines is performed by analyzing data collected by sensors, which are located on machine components and also in production environments. The collected data record real-time situations and reflect the correctness of mechanical system conditions. When the tendency of a mechanical failure emerges, experienced operators in factories are able to take appropriate operations to prevent the outage situations of production systems.

As today's manufacturing market is becoming more competitive, how to improve the availability, sustainability, and quality of manufacturing services in smart factories has been a crucial concern for manufactures. This situation has triggered the demand for implementing predictive maintenance on production lines, which refers to the maintenance activities that are performed to avoid failure occurrences and to improve the availability and safety of the maintained system. Within manufacturing processes, the detection of anomalies such as mechanical faults and failures enables the launching of predictive maintenance tasks, which aim to predict future faults, errors, and failures and also enable maintenance actions. Predictive maintenance is an important task by which the machine or mechanical system deterioration tendency and the location of a failure can be detected.

### 4.4.1 Predictive maintenance in industry

In industry, a predictive maintenance task relies on the monitoring of a measurable system diagnostic parameter, which identifies the state of a system [GDBR02]. According to the current state of a machine, if any fault or failure exists, a diagnosis can be launched to determine the causes of the fault or failure. Also, based on the characteristics of the fault or failure, analysis about how they will propagate and evolve over time can be performed. In this way, machine or mechanical system deterioration tendency and the location of a failure can be predicted. Based on the prediction results, maintenance decisions, such as calling the intervention of a machine operator are proposed according to the severity of anomalies, to prevent the halt of the production lines and to minimize economic loss. The use of predictive main-

tenance techniques has several advantages, such as improved machine availability, improved production efficiency, and reduced maintenance cost [WY07, Rao96].

Normally, when a propensity of machinery fault or failure is detected, highly experienced machine operators are capable of performing appropriate actions to prevent the outage situation of the production system. However, as the structure and behavior of production systems are getting more and more complex, the volume of machine operating data grows significantly. Thus it is possible that the domain professionals fail to respond to a machinery fault or failure timely and accurately. For this reason, manufacturing companies are searching for solutions through which they can manage this big data efficiently and perform prognostics tasks intelligently. Several techniques have been used to detect wear and tear in mechanical units and to predict future machinery conditions, such as machine learning, data mining, statistics, and information theory [KOK+17, LHT17, KITT17].

Recently, these techniques are integrated to construct a CPS, which is the central technique of Industry 4.0. As the data generated by sensors and networked machines gets higher in volume, a CPS is equipped with advanced technologies to obtain the capabilities of self-awareness and self-maintenance. These capabilities contribute significantly to the resilience, automation, and productivity of the CPS, which ensures the CPS to propose predictive decisions in an intelligent and optimal manner. Because of these benefits, CPS-based predictive maintenance has become a promising approach to detect and predict anomalies in manufacturing processes [LBK15].

Normally, a CPS-based predictive maintenance task is performed through a three-layer collaboration. Fig. 4.4 shows the collaboration among different layers and the architecture of a CPS-based predict maintenance system. We start with describing the physical space.

### 4.4.2 The physical space

The physical space is at the bottom layer of the architecture. At this layer, machine operating data is gathered using sensors located on the machines and tools. The sensors measure system diagnostic parameters such as vibration, wear, length, shape and temperature measurement, etc. Additional information are collected from the manufacturing environment, the product itself as well as the machine operator's experience.

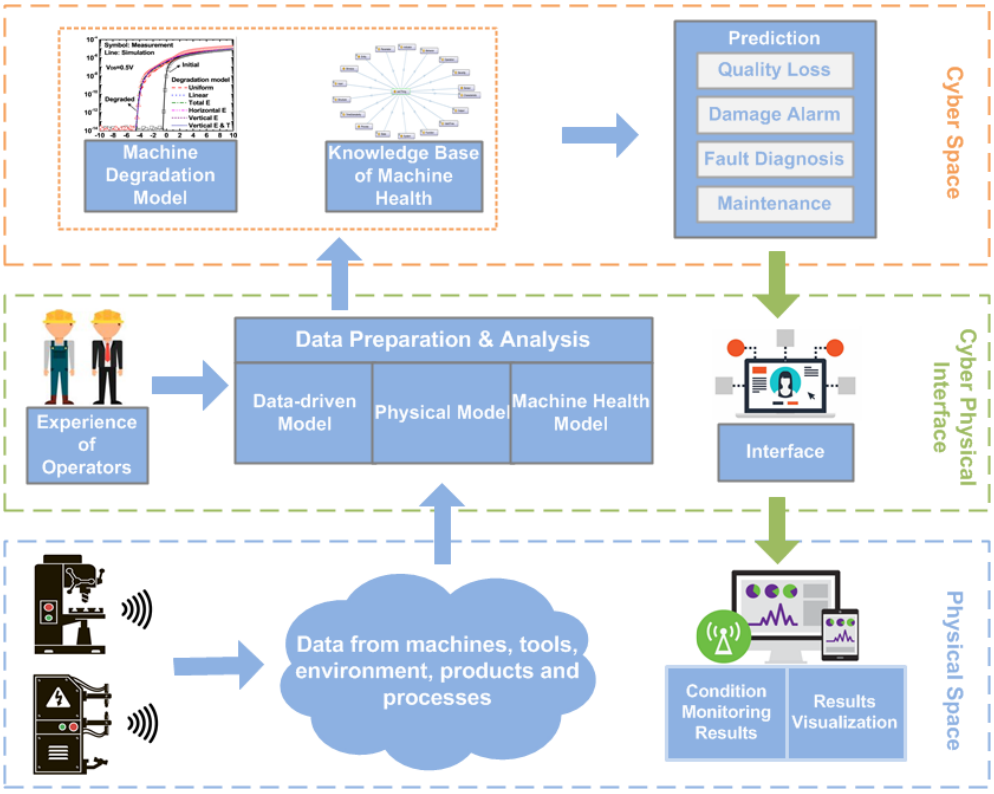



Figure 4.4 – The predictive maintenance task based on a cyber-physical approach.

Additionally, machine historical data are collected over time. These data are used to learn about the past behavior of machines and to extract predictive knowledge. To ensure data integrity and confidentiality, notable research efforts have been paid to provide lightweight cryptographic mechanisms [TS07, WKM04]. These mechanisms have significantly improved the security of CPSs [CAS+09].

### 4.4.3 The cyber-physical interface

The collected sensor data are stored and processed at the intermediate layer, named the cyber-physical interface. At this layer, statistical techniques such as machine learning and data mining algorithms are used to process the collected data in order to understand the manufacturing processes and to learn from operators' experience. Also, big data techniques are implemented to ensure the efficient integration and analysis of manufacturing data.

For the goal of predictive maintenance, valuable information is extracted from the big industrial data, and it is used to establish a relationship between nominal behavior and the current state of operation of machines [NB17]. By this, the cor-

rectness of the current machine condition (normal or abnormal conditions) can be inferred, which allows maintenance actions to be scheduled.

### 4.4.4 The cyber space

The cyber space provides decision-making about machine failure prognostics and maintenance. At this layer, the virtual profiles of machines such as machine footprints are connected to form a machine network. After that, machines can register their virtual profiles into the network and exchange information within the cyber space.

To ensure successful data and information exchange, it is necessary to have unified data analysis tools to process diversified industrial data. The diversified industrial data include structured, semi-structured, and other unstructured data formats. Within the cyber space, the data formats are converted to uniformed standards, to improve the efficiency of data storage, query, retrieval, processing, and analysis [ZQT+15].

On the other hand, advanced methods such as the Time Machine are developed to track the changes of machine status, infer additional knowledge from historical information, and apply peer-to-peer comparison among machines [LBK15].

Once the machine network is constructed, information is continuously pushed from every machine to the machine network. At this layer, machine degradation models are extracted from the mining of sensor data. Also, a knowledge base of machine health is created to represent the machine condition at the cyber space. By this, the health condition of machines can be accurately simulated and appropriately presented to stakeholders upon their demand without geographical limitations [LBK14]. Leveraging the machine degradation models and the knowledge base, the CPS is capable of providing precise machine health predictions and required maintenance decisions. After that, the predictive system uses these prediction results and maintenance decisions to propose timely maintenance and avoid downtime of manufacturing processes.

## 4.5 Summary

In this chapter, we have given the background knowledge of Industry 4.0 predictive maintenance. We first introduced the four key components of Industry 4.0: the IoT,

CPS, Cloud Computing, and Big Data analysis technologies. We then presented the 5C architecture of a CPS, which is the core technique of Industry 4.0. At last, we described a predictive maintenance task in the context of Industry 4.0, which is based on a cyber-physical approach. The approach is accomplished through a three-layer collaboration between the physical space and the cyber space.

# Chapter 5

# Existing models for industrial predictive maintenance

## Contents

## 5.1 Introduction

Over the last decades, a considerable amount of research efforts have been undertaken to address the development of different models for industrial predictive maintenance. For a predictive maintenance task, appropriate model selection is vital to manufactures. A proper selection of a model requires not only a thorough mathematical understanding of it but also the knowledge of how to implement it in real-world scenarios [SHM11].

In this chapter, we classify the existing models for industrial predictive maintenance into four categories: i) the knowledge-based models; ii) the physical models; iii) the data-driven models; and iv) the hybrid models. We also present the advantages and disadvantages of each type of models.

## 5.2 Knowledge-based models

This type of models access the similarity between an observed situation and a database of previously defined failures, and then deduce the future failures or the life expectancy from previous events [SHM11]. The knowledge-based models can be further categorized into expert systems and fuzzy systems.

### 5.2.1 Expert systems

An expert system is a computer program designed to simulate the problem-solving behavior of an expert in a narrow domain or discipline [Nad13]. Normally, this type of system consists of a knowledge base that contains accumulated knowledge acquired from domain experts and a rule base for applying acquired knowledge to particular problems. A rule base contains a set of IF-THEN-based logical rules, which are formulated based on heuristic facts acquired from previous system behavior or domain experts. An inference engine is another key component of the system that applies logical rules to the knowledge base to deduce new information. Expert systems are also called knowledge-based systems [Nad13].

There are several advantages of expert systems. Firstly, expert systems use explicit representations of knowledge in the form of words and symbols. This eases the understanding of the knowledge by a human, comparing to the numerically derived models in computational intelligence [JJ90]. Secondly, since knowledge-based systems contain explicit semantics of the domain, it helps to solve the semantic gap

issue, which stands for the incoherence between the knowledge extracted from industrial data and the interpretation of the knowledge from a user [DWL15]. Thirdly, knowledge-based techniques ensure the semantic interoperability among different systems. Since the production systems in industry are becoming more and more complex, it is not easy to integrate heterogeneous resources that are developed with different vocabularies and different perspectives on the data [HH00]. To address this issue, knowledge-based systems ensures the data exchange with formal and rich semantics. This provides the precise meaning of the data to different systems, which eases the interpretation and management of the data.

However, expert systems suffer from two main disadvantages. Firstly, when there is no domain experts available, it can be difficult to define a comprehensive set of logical rules. Moreover, since the prediction outputs are determined by a discrete set of rules, traditional expert systems are not feasible to be used for a continuous variable prediction [SHM11]. To cope with this issue, advanced technologies such as stream reasoning should be adopted to deal with real-time and continuous data [DDVvHB17]. In stream reasoning, continuous queries and rules are used to reason on real-time data streams.

### 5.2.2 Fuzzy systems

For the classic logic used in an expert system, a statement can be either true or false, which means a piece of data is classified either inside or outside of a set. However, to solve a real-world problem, sometimes it is not necessary to define a membership with such precision. In this context, fuzzy systems are developed, within which the IF-THEN-based logical rules are intentionally made imprecise. This type of systems use fuzzy set theory to overcome the deficiency of traditional expert systems by assigning partial set membership to data based on their 'degree of truth' [SHM11]. In this way, input data can belong to a fuzzy variable with a certain degree, allowing it to be part of more that one fuzzy variable at the same time. This fuzzification approach assures a better description and expression of imprecise knowledge [BT97].

Fuzzy systems are feasible to provide results when the input data is imprecise and incomplete. Also, compared to expert systems, fuzzy systems require less number of rules for the reasoning process. These advantages ensure them to have better efficiency and easier implementation than expert systems. However, fuzzy systems rely heavily on domain experts to define the fuzzy variables and to specify the fuzzy

rules, thus making them hard to be adopted when no domain experts are available.

## 5.3 Physical models

Physical models use explicit mathematical representation to formalize the physical understanding of a degrading machine or equipment [Pec10]. This type of models address a predictive maintenance task by solving a deterministic equation or a set of equations derived from extensive empirical data [SHM11]. By using physical models, predictive maintenance of a physical asset is achieved through the interpretation of the acquired knowledge of a manufacturing process, and through the analysis of possible hazards that may cause a failure. Normally, the main steps of a physical model-based predictive maintenance approach include failure modes and effects analysis (FMEA), feature extraction, and remaining useful life (RUL) estimation [Pec09].

The advantages of physical models are their capability of direct incorporation of existing physical mechanisms that have been proved and well-understood by extensive and exhaustive empirical testing [SHM11]. As the physical understanding of a monitored system improves, a physical model can update itself to improve its accuracy and quality. Furthermore, changes in the model outputs are described by the residuals (the differences between the reality and the model), which normally have a direct and translatable physical meaning. This eases the interpretation of the outputs of a physical model [SHM11]. Because of these advantages, physical models have been widely used in the predictive maintenance of different physical assets, such as air vehicles [RNB01], turbine engines [HZTM09, SHM11], and military systems [EPMC12], etc..

The main disadvantage of a physical model is the difficulty of assigning appropriate parameters used in the model. Since the massive and multivariate data required for the assignment of parameters are usually not available, it is hard to quantitatively characterize the system behavior [SHM11]. Also, since fault and failure mechanisms may vary from one equipment to another, this type of models are normally equipment-dependent. Thus it is hard to identify the fault and failure mechanisms of a newly monitored machine component/machine without interrupting system operation [HZTM09].

## 5.4 Data-driven models

In recent years, rapid advances have been made in the research of data-driven models and approaches for accurate predictive maintenance. These models perform like a black box that learn the behavior of physical assets directly from their operation data [JGZ17]. Within a data-driven approach, knowledge about machines are extracted internally from machine operation data, instead of externally from domain experts. Normally, data-driven approaches are classified into machine learning techniques and statistical techniques [DGD+09, PDZ10].

### 5.4.1 Machine learning-based approaches

Recently, Machine learning (ML) approaches have been proven to be effective solutions for predictive maintenance. The implementation of ML models have been facilitated by the growing capabilities of hardware, cloud computing, and newly introduced state-of-the-art algorithms [SSP+14].

Depends on the characteristics of data, the learning process of ML-based predictive maintenance can be categorized into three types [JGZ17]:

- Supervised learning which is applied to labeled data. This type of methods aim to learn a function that maps an input to an output based on a set of labeled training examples. Typical learning algorithms for supervised learning are Support Vector Machine (SVM) [SS01], Artificial Neural Networks (ANN) [JMM96, JGZZ11], Logistic Regression [KDG+02], Naive Bayes [R+01], Random Forests [LW+02], and Decision Trees [Qui86], etc.

- Unsupervised learning applied to unlabeled data. In unsupervised learning, the training data is not associated with any corresponding target values. In other words, no labels are given to the learning algorithm, leaving it on its own to discover patterns and structure from the input data. The objective of an unsupervised learning process is to segregate data points with similar traits and assign them into clusters, which is known as *clustering*, or to construct an estimate of the distribution of the input data, which is known as *density estimation*. Common unsupervised learning methods include k-means Clustering [LVV03], Gaussian Mixture Models [Rey15], Self-organizing Maps [Koh97], etc.

- Semi-supervised learning applied to both labeled and unlabeled data. This type of learning methods is a learning paradigm concerned with the study of how computers and natural systems such as humans learn in the presence of both labeled and unlabeled data [ZG09]. In addition to unlabeled data, semi-supervised learning algorithms are provided with a certain level of supervision, but not for all the training examples. Semi-supervised learning requires less human effort than supervised learning, meanwhile giving higher accuracy than unsupervised learning [ZG09]. Because of these advantages, it has been pervasively implemented in real-world practices. Typical techniques used in this field are Self-Training algorithms [RHS05], Generative Models [KMRW14], Semi-supervised Support Vector Machines [BD99], Graph-Based Algorithms [Bry86], and Multi-view Algorithms [BS04], etc.

Facilitated by the capabilities of the advanced techniques and state-of-the-art algorithms, machine learning-based approaches have shown to be promising and effective solutions for industrial predictive maintenance. The algorithms and techniques introduced in this subsection have been widely used in real-world practices to reduce the maintenance cost and production downtime [SSP+14].

### 5.4.2 Statistical learning-based approaches

Within this kind of approach, predictive maintenance is achieved by fitting the empirical model (a function) as close as possible to the collected data and extrapolating the fitted curve to failure criteria [JGZ17]. A typical statistical model is a regression method for trend extrapolation, which is based on linear, exponential, or logarithmic functions. The common methods used in statistical learning-based approaches are i) Stochastic filtering [Kal13]; ii) Particle filters and their variants [GGB+02]; iii) Hidden Markov models [BGR02]; iv) Time series analysis [Ham94].

Because of the low cost of deployment and better applicability than physical models, data-driven models have been widely applied to industrial predictive maintenance. However, the main disadvantage of this type of model is their demand for higher volume data than physical models. To well train a data-driven model, sufficient run-to-failure data ("sufficient" quantity means that data have been observed for all fault modes of interest [UGL08]) needs to be collected for the model to capture complex relations among data [JGZ17]. This means a large amount of machine historical data needs to be collected ahead of time for obtaining high-quality and

accurate data-driven models.

## 5.5 Hybrid models

A hybrid model applies both physics-based and data-driven approaches. As mentioned in the previous subsection, data-driven models are feasible to be used when the required big data is easy to collect. However, it is usually the case that only part of machine historical data can be obtained [Wan16]. In this context, the data-driven approach is jointly used with the physics-based approach for effectively identifying machine conditions. Existing hybrid model-based predictive maintenance approaches can be classified into two types: i) Series approaches and ii) Parallel approaches.

In series approaches, physical models are combined with online parameter estimation techniques to update model parameters when new data are available [JGZ17]. Data-driven methods are used to tune the parameters of physical models. This type of approaches have been applied to the predictive maintenance of various equipments, such as power metal-oxide-semiconductor field-effect transistors (MOSFETs) [CSSG11], printed circuit card assemblies [Pec10], and Lithium-ion (Li-ion) batteries [SCR+12], etc.

Within parallel approaches, data-driven models are trained to predict the residuals not explained by the first principle model [JGZ17]. Normally, the hybrid model used in a parallel approach is created with an individual approach, which is either physics-based or data-driven. Therefore, the accuracy of a parallel hybrid model is normally higher than a series hybrid model [JGZ17]. However, implementing a parallel hybrid model requires several steps, which leads to a higher modeling complexity than series approaches-based models. Because of this, parallel hybrid models consume more computational time than series hybrid models.

In general, the different steps for implementing a parallel hybrid model for predictive maintenance are: parameter identification, condition monitoring, feature extraction, healthy baseline creation, anomaly detection, parameter isolation, failure definition, parameter trending, and RUL estimation [CP09].

## 5.6 Summary

Predictive maintenance is a key technique implemented in smart factories to improve the availability, reliability, and productivity of manufacturing systems. In this chapter, we have reviewed the existing models and approaches for industrial predictive maintenance by classifying them into four categories: knowledge-based models, physical models, data-driven models, and hybrid models. The classification was followed by the demonstration of the advantages and disadvantages of the existing models.

We use Fig. 5.1 to summarize our classification of the common predictive maintenance approaches, where the existing approaches are structured into four levels, marked with different colors. We first mark the common approaches with a super class (the rectangle with blue color), and then further divide the super class into four sub-classes (rectangles with green color). After that, the more specific models are presented by the rectangles with yellow color. At last, the specific models are are further classified into domain-specific techniques, which are represented by the rectangles with gray color.

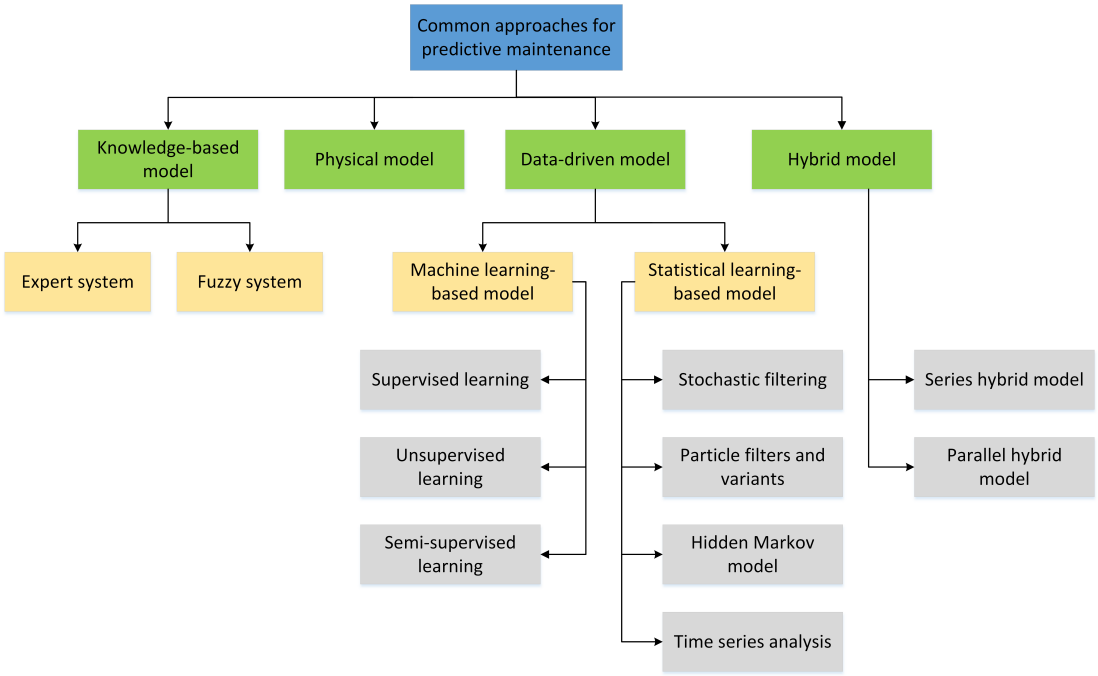



Figure 5.1 – Classification of the common predictive maintenance approaches.

Since the industrial domain is becoming more and more knowledge-intensive, it is always beneficial to incorporate domain knowledge into the production pro-

cess. Also, reasoning capabilities are required for the predictive maintenance system to predict failures and propose maintenance decisions automatically. In this context, we pay special attention to the knowledge-based models/systems. This type of models/systems emulate the performance of a human expert in a specific domain for solving real-world problems. In predictive maintenance, a knowledge-based model/system uses knowledge management and knowledge representation techniques (e.g., semantic technologies [DFH11]) to promote the prediction and diagnosis of machinery failures and to propose intelligent maintenance strategies. In this thesis, we develop knowledge-based predictive maintenance systems to address the open challenges mentioned in Chapter 3.

# Chapter 6

# Knowledge-based systems: theoretical foundations

## Contents

## 6.1 Introduction

A *knowledge-based system* maintains a knowledge base which stores the symbols of a computational model in form of statements about the domain, and performs reasoning by manipulating these symbols [Gri09]. As introduced in Chapter 5, knowledge-based systems use explicit representations of knowledge in the form of words and symbols, which eases the understanding of the knowledge by a human or a computer. Also, they help to solve the semantic gap issue and ensure the semantic interoperability among different systems and users.

Nowadays, organizations and companies in the manufacturing domain are becoming increasingly collaborative and knowledge-intensive [OBGM11]. However, large part of valuable knowledge are hidden and not directly available to the users. Recently, many companies and organizations have realized the importance of capturing, structuring, representing, and reuse knowledge for coming up with intelligent decision makings [Lei10]. In this context, knowledge-based systems are powerful tools that use knowledge techniques to solve complex real-world problems.

In this thesis, we aim to develop a knowledge-based system to facilitate the predictive maintenance tasks. In this chapter, we introduce the theoretical foundations and basic concepts of knowledge-based systems.

## 6.2 Knowledge-based systems: the classic architecture

Normally, the classic architecture of a knowledge-based system consists of four basic components. We use Fig. 6.1 to show the interaction among different components. The four basic components are:

- A *knowledge base* that contains a collection of information and knowledge for a certain domain.
- An *inference engine* that provides inference capabilities to deduce insights from the information and knowledge that is structured in the knowledge base.
- A *working memory* that holds case-specific data/facts about the initial problem and intermediate inference results.
- An *interface* to the outside world that allows other computer systems or users to interact with the knowledge-based system.

By using the classic architecture, a knowledge-based system imitates the decision making process of a human brain, within which the new knowledge and experience are continuously updated in the knowledge base, while the control process (inference engine) remains unchanged in their nature. Within a knowledge-based system, the knowledge is explicitly structured and represented in the knowledge base, rather than implicitly in the structure of a program. Because of this advantage, domain experts who may not have programming expertise can easily manipulate and update the knowledge in the knowledge base.

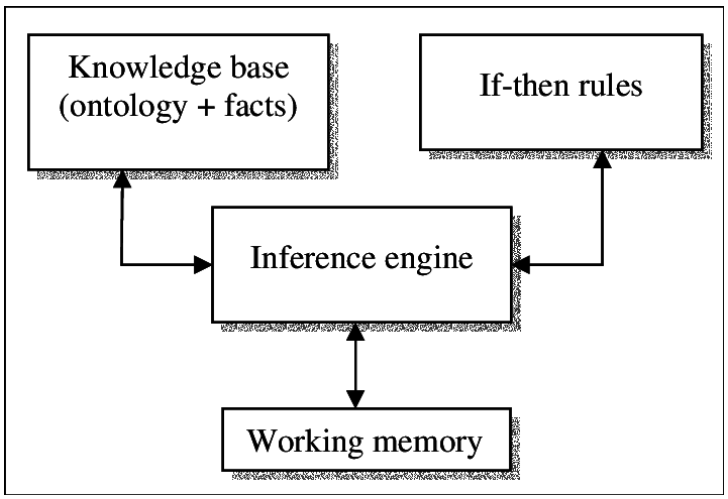



Figure 6.1 – The classic architecture of a knowledge-based system [SS08].

According to different problem-solving objectives and approaches, the encoding methods of knowledge in a knowledge-based system may also change. Some knowledge-based systems encode expert knowledge in the form of logical rules, which are referred to as rule-based systems [Ber88]. The main advantage of rule-based systems lies in the easy and straightforward encoding of knowledge. Since logical rules are generally written in a form that closely resembles natural language, they are easy to be understood by novice users that do not have sufficient domain knowledge.

Some knowledge-based systems use case-based reasoning techniques. A case-based reasoning approach replaces logical rules with cases, within which the cases are essentially the solutions to existing problems and can be used to solve new problems [Kol14]. Case-based reasoning provides inference capabilities based on previous experience. This allows a system to recall the solutions of similar past problems and adopt the right strategy to solve new problems.

However, the knowledge-based systems that use classic architecture do not always provide satisfactory results when solving real-world problems [ZM15]. The main drawback of the classic architecture is that the conceptual model obtained from the knowledge elicitation process is often incomplete [CWAM75]. Using a non-complete conceptual model for reasoning may lead to the failure of inferring correct knowledge (e.g., when real-time data does not match with any of the pre-defined logical rules, a predictive maintenance system may fail to predict a machinery failure before the failure happens in the near future). To deal with this issue, the collection of experience from domain experts is required. In this context, complementary approaches that provides the capability of *experience capitalization* needs to be added to the classic architecture [ZMMZWdB15].

## 6.3 Knowledge-based systems: the KREM architecture

To address the drawbacks discussed in the previous subsection, a new modular architecture for knowledge-based systems has been proposed recently, which is called KREM (Knowledge, Rules, Experience, and Meta-Knowledge) [ZM15, ZMS19]. The main advantage of the KREM architecture is its complement to the classic architecture with experience capitalized from domain experts and meta-knowledge. In this way, the KREM architecture improves the level of completeness and flexibility of knowledge-based systems.

There are four components in the KREM architecture [ZM15]:

- The *Knowledge* component. It contains domain knowledge to operate. Normally, the domain knowledge is structured and represented by formal conceptual models, such as ontologies [Gru09]. We will introduce the concept of ontologies in Section 6.4.1.

- The *Rules* component. Inside this component, different reasoning capabilities (monotone, spatial, temporal, spatial–temporal, fuzzy, etc.) are supported.

- The *Experience* component. It allows the capitalization of experience and reuse of previous knowledge. This component aims to complete the knowledge model incorporated in the *Knowledge* component and the rule base formulated in the *Rules* component.

- The *Meta-knowledge* component. This component includes knowledge about the other three bricks with regard to specific problems.

Because of its modular architecture, the KREM model has been adopted as a design pattern for various knowledge-based systems [STV+07, BPERFM09, KS11, CGZM+19, TSC+12]. In the KREM model, each component has specific functionality and is separated from each other. This design pattern ensures that a change inside one component does not affect the operation of other components. This advantage facilitates the reusability and flexibility of the KREM architecture [ZM15].

In this thesis, the KREM architecture is used as a design pattern for developing our intelligent predictive maintenance system. The development and implementation of different knowledge components will be demonstrated in Part II of the thesis.

## 6.4 KREM architecture: the *Knowledge* component

Within the *Knowledge* component of the KREM architecture, a formal conceptual model performs a central role for knowledge representation and reuse. Among the existing conceptual models, semantic technologies, especially ontologies, have been widely used to formalize domain knowledge.

### 6.4.1 Concept of ontology

The term *ontology* originated in philosophy. It studies concepts that directly relate to being, in particular becoming, existence, reality, as well as the basic categories of being and their relations [Sim99]. In computer science, an ontology is considered as "*an explicit specification of a conceptualization for a domain of interest* [Gru93]. Within this definition, *specification* refers to an act of describing or identifying something precisely. This requires the concepts and relationships in ontolgies to be clearly defined by using formal logic. *Conceptualization* stands for an intentional semantic structure that encodes implicit knowledge constraining the structure of a piece of a domain [Aya18]. Normally, the conceptualization within an ontology is formalized by a logic theory that is written in a certain language. Also, ontologies provide reasoning capabilities, by which new knowledge can be inferred.

Since ontologies are developed based on formal logic foundations, they have been pervasively used in industry to ensure the semantic interoperability among different systems and users. In the predictive maintenance domain, ontologies

play a key role in many distributed intelligent system as they provide a shared, machine-understandable vocabulary for information exchange among dispersed agents [AUM12]. Large ontologies are designed in a modular structure, to enhance their reusability, extendability, and easy maintenance.

In the following subsections,the ontology-related background knowledge is introduced, including ontology components, common ontology development methods, development languages, development tools and ontology reasoning techniques.

### 6.4.2 Ontology components

Normally, there are four building blocks that construct an ontology:

- *Individuals*, which represent objects in the domain in which we are interested. Fig. 6.2 shows an example of a set of individuals that represent the concept of *animal*. In the figure, individuals are marked as a cross symbol.

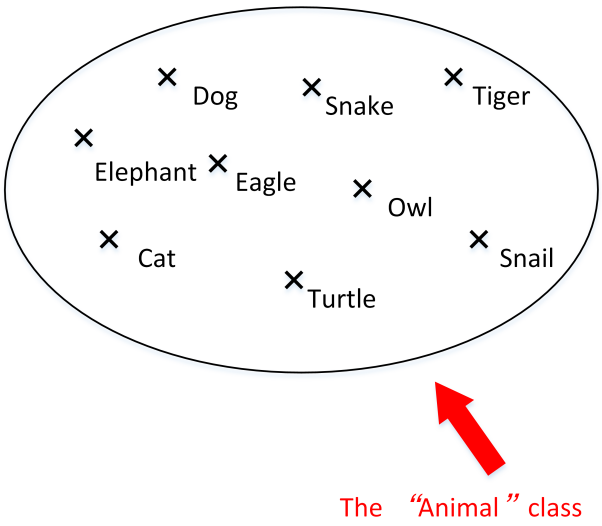



Figure 6.2 – An example set of individuals.

- *Classes*, which provide an abstraction mechanism for grouping resources with similar characteristics. They are considered as sets that contain individuals. Normally, classes are described using formal (mathematical) descriptions that state precisely the requirements for membership of the class [HJM+09]. Classes may be organized into a superclass-subclass hierarchy, for example, the class *Mammal* can be a subclass of the class *Animal*. In this case, *Animal*

is the superclass of *Mammal*. Fig. 6.3 shows a group of classes that contain individuals. There are three classes in the figure: *Person*, *Animal*, and *Vegetation*. They are represented by ellipses, with individuals inside them.

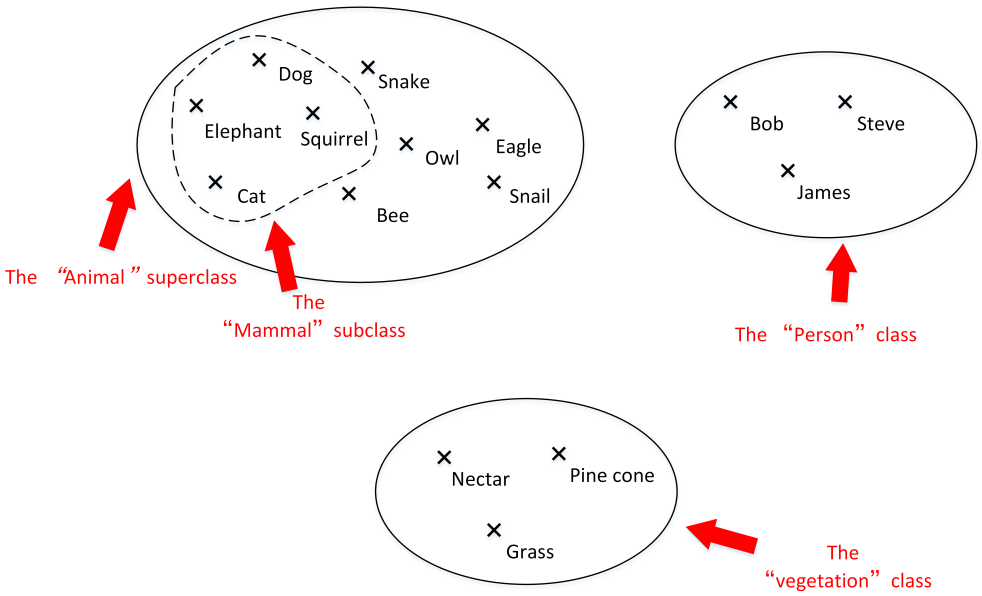



Figure 6.3 – An example of three classes: *Person*, *Animal*, and *Plant*.

- *Properties*, which are binary relations among individuals. *Properties* are used to link individuals together, and to describe the features and attributes of them. *Properties* are categorized into two types: *Object Properties* that link two individuals, and *Data Properties* that link individuals to data values. Fig. 6.4 shows a set of *Properties* (arrows) that link individuals under different classes. The names of these *Properties* are assigned beside the arrows.

- *Restrictions*, which describe a class of individuals based on the relationships that members of the class participate in.

After introducing the four core components of an ontology, we give the formal definition of an ontology [BDPH06]:

**Definition 1** (**Ontology**). *An ontology is a quadruplet* $O :< C, P, Sub, Applic >$, *where:*

- C *is the set of the classes used to describe the concepts of a domain of interest.*

- P *is the set of properties used to describe the individuals of the* C *classes.*

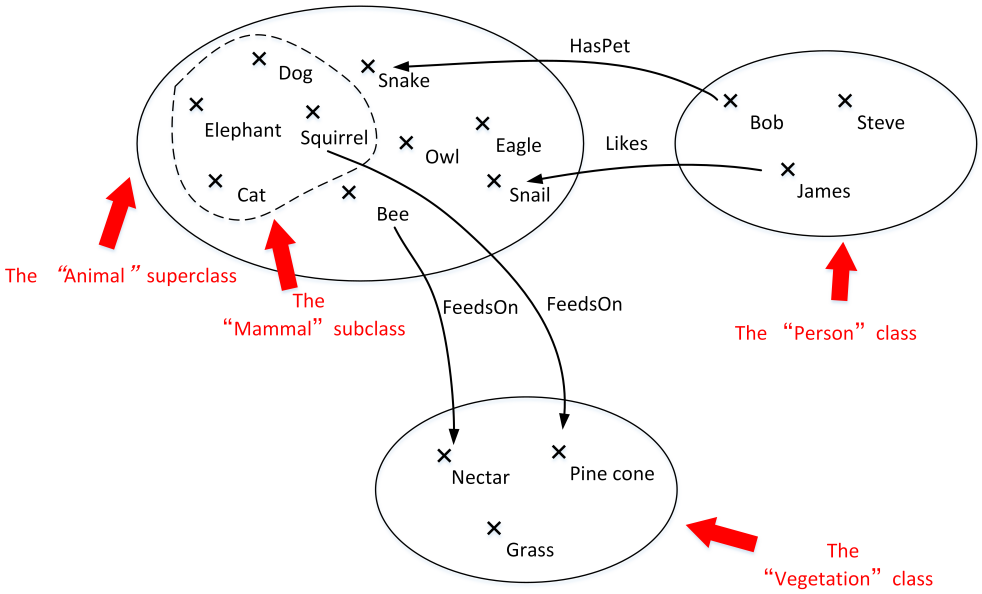



Figure 6.4 – Properties among different individuals.

- $Sub$ *is the subsumption relation function defined as* $Sub : C \rightarrow 2^C$, *where* $2^c$ *denotes the power set of* C. *For a class* $c_i$ *in the ontology, it associates its direct subsumed classes. Class* $C_1$ *subsumes class* $C_2$ *iff* $\forall x \in C_2, x \in C_1$.

- $Applic$ *is a function defined as* $Applic : C \rightarrow 2^P$. *This function associates to each ontology classes to those properties (object properties and data properties) that are applicable for each instance of this class.*

In chapter 8, we demonstrate the ontological framework that we developed for the knowledge-based predictive maintenance system. Within the ontological framework, main classes, properties, and their restrictions are introduced.

### 6.4.3 Ontology development methodologies

There are a number of methodologies that address the issue of ontology development and maintenance. In this subsection, we give a brief review of the commonly used methodologies. These methodologies include: TOVE [GF94], the Enterprise Model Approach [UK95], METHONTOLOGY [FLGPJ97], IDEF5 [PMM+94], and CommonKADS [SWdH+94]. In the next subsections, these methodologies are introduced in detail.

### The TOronto Virtual Enterprise (TOVE) methodology

The TOVE methodology was proposed during the development of ontologies for enterprise engineering. It consists of six main steps [GF94]:

1) *Motivating scenarios.* It is the start point where a set of problems encountered by a particular enterprise is clarified and described. Normally, the problems are described in the form of stories or examples.

2) *Informal competency questions.* These questions are the requirements of the ontology to be developed, which are proposed based on the motivation scenarios. They are in a form of informal questions that an ontology must be able to answer. This step is also considered as an evaluation on the ontological commitments that are made in the previous step.

3) *Terminology specification.* Within this step, the objects, properties, and attributes are formally specified. Normally, the specification is done by first order logic [Smu95].

4) *Formal competency questions.* The requirements of the ontology (proposed in step 2) are formalized in a form of formally defined terminologies.

5) *Axiom specification.* Axioms that specify the definition of terms and constraints are given in first order logic. During this step, the axioms must be necessary and sufficient to express the competency questions and their solutions.

6) *Completeness theorems.* This step acts as an evaluation stage which assesses the competency of the ontology. The evaluation is done by defining the conditions under which the solutions to the competency questions are complete.

The main advantage of the TOVE methodology is its capability for ontology evaluation, with regard to the completeness theorems. This advantage facilitates its use for assessing the extendibility of an ontology, for example, any extension of the ontology must also validate the completeness theorems [GF94].

### The Enterprise Model Approach

The Enterprise Model Approach was proposed based on the experience of developing enterprise ontologies. After several rounds of improvement, the approach has been refined into four main steps [UK95]:

1) *Identify purpose.* This step determines the level of formality at which the ontology should be described.

2) *Identify scope.* This step aims to outline the range of information and knowledge that an ontology needs to characterize. This step can benefit from the motivating scenarios and informal competency questions (in the TOVE methodology), such as producing a list of relevant concepts and relations by brainstorming, after that deleting the irrelevant and synonymous ones.

3) *Formalization.* Formal definitions and axioms are created according to the terms produced in step 2.

4) *Formal evaluation.* Use general criteria to evaluate the ontology, or check against the competency questions.

**METHONTOLOGY**

Different from the previous two methods, METHONTOLOGY follows project management techniques and produce an ontology as a finality of the process. It starts the ontology development process by identifying a set of activities [FLGPJ97]:

1) *Specification.* Identify the purpose of the ontology, such as the intended users, the degree of formality required, etc.. After that, determine the scope of the ontology including the set of terms to be represented, their characteristics and the required granularity. The result of this step is a natural language-based ontology specification document.

2) *Knowledge acquisition.* This step aims to extract and elicit knowledge from heterogeneous knowledge sources, such as expert interviews, plain texts, or online resources.

3) *Conceptualization.* Identify the domain terms as concepts, relations, instances, and properties. The representation of the terms is in an informal way.

4) *Integration.* The objective of this step is to ensure the uniformity of conceptualization across ontologies. To achieve this goal, definitions of a concepts from other ontologies should be incorporated.

5) *Implementation.* Within this step, the ontology is represented in a formal language.

6) *Evaluation.* This is one of the main steps in the METHONTOLOGY approach. Techniques for the validation and verification of knowledge-based systems are implemented. During validation and verification, the common quality issues of knowledge-based systems are i) incompleteness, ii) inconsistency, and iii) redundancy [JBCV98].

7) *Documentation.* The documents generated from the previous steps should be collected and maintained.

Similar to TOVE, the most distinctive aspect of METHONTOLOGY is the maintenance of ontologies. Compared to TOVE, METHONTOLOGY proposes the idea of *evolving prototype* during the ontology life cycle. In this way, it addresses more maintenance issues throughout the maintenance stage [FLGPJ97].

**The Ontology Description Capture Method (IDEF5)**

IDEF5 methodology covers a wide range of ontology engineering topics, such as creation, modification and maintenance [PMM+94]. This methodology is a general procedure for developing ontologies, which includes five main steps:

1) *Organizing and scoping.* In this step, the purpose, viewpoint, and context for ontology development is clarified. *Purpose* includes objectives and requirements, and *scope* defines the boundaries of the ontology.

2) *Data collection.* Raw data is acquired for ontology development. Traditional knowledge acquisition techniques are used in this step, such as protocol analysis and expert interview.

3) *Data analysis.* The ontology is extracted from the data collection results. Objects are listed, followed by the identification of the boundaries of the ontology.

4) *Initial ontology development.* A preliminary ontology is developed, including initial descriptions of concepts, relations and properties.

5) *Ontology refinement and validation.* The developed ontology is iteratively refined and tested. As the ontology is instantiated with real data, the results of instantiation are compared with the original ontology structure.

The key distinguishing characteristic of IDEF5 is the gradual refinement process during ontology development. This advantage allows the developed ontology to acts as an evolving prototype model, which ensures the ontology is progressively enriched with real data while following its original structure.

#### CommonKADS

CommonKADS is a widely used methodology for developing knowledge-based systems, where ontologies play a central role [SWdH+94]. Instead of merely focusing on how to *extract* knowledge from experts and transfer it to machines in a computational form, the CommonKADS methodology provides a structured development approach for the *modeling* of knowledge-based systems. This modeling methodology not only takes expert knowledge into account but also investigates how that expert knowledge is embedded and used in the organizational environment.

Following the CommonKADS methodology, a suite of templates for problem-solving models were proposed, including Organization Model, Expertise Model, Task Model, Agent Model, Design Model, and Communication Model. These formal models are jointly used with project management and planning techniques to serve as guidelines for the development of knowledge-based systems. By this, they provide mechanisms for flexible project configuration and control [SWdH+94].

### 6.4.4 OWL: Web Ontology Language

The Web Ontology Language (OWL) is developed by the World Wide Web Consortium (W3C) as a formal ontology language. It is a component of Semantic Web that used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. The representation of terms and their interrelationships form an ontology.

As a key language in the Semantic Web stack, OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics [MVH+04]. To do this, OWL is proposed based on Resource Description Framework (RDF) and Resource Description Framework Schema (RDFS), which are layered on the top of the Extensible Markup Language (XML). Fig. 6.5 shows the hierarchy of these Semantic Web languages, within the Semantic Web stack. Using this layered architecture, OWL makes easier for machines to automatically process and integrate

information available on the Web.

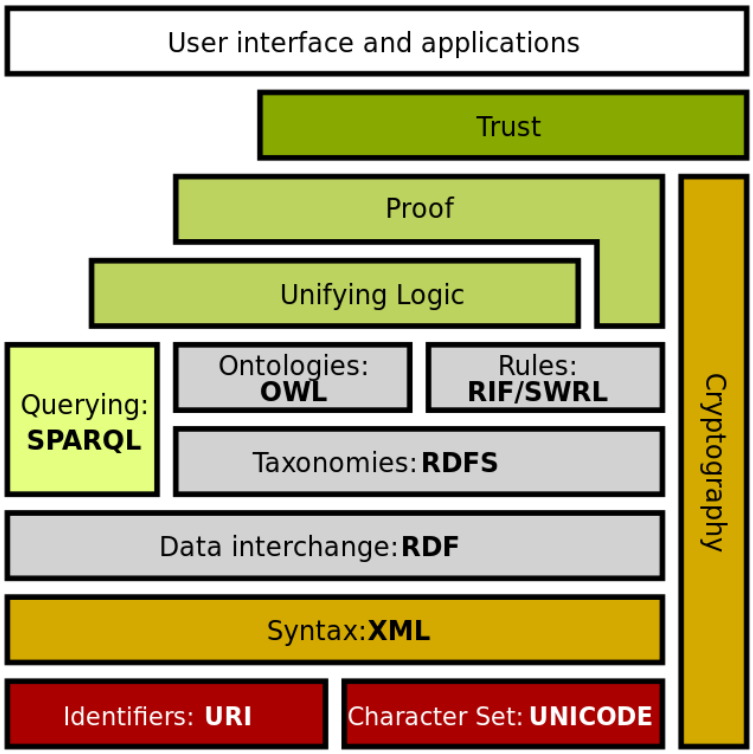



Figure 6.5 – The Semantic Web stack. [1]

OWL provides three increasingly expressive sublanguages, they are [MVH+04]:

- *OWL* Lite, which supports users with their primary needing of a classification hierarchy and simple constraints. Owl Lite has a lower formal complexity than OWL DL, thus providing a quick migration path for thesauri and other taxonomies.

- *OWL DL*, which supports users who want the maximum expressiveness while retaining computational completeness and decidability. OWL DL includes all OWL language constructs, but they can be used only under certain restrictions (e.g., while a class may be a subclass of many classes, a class cannot be an instance of another class).

- *OWL Full*, which is designed for users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. For example, in OWL Full a class can be treated simultaneously as a collection of indi-

[1] https://commons.wikimedia.org/wiki/File:Semantic_web_stack.svg

viduals and as an individual in its own right. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary.

### 6.4.5 Ontology development tools

In the last decade, several software tools for developing and manipulating ontologies have been proposed by researchers in the semantic web community. Most of these contributions were addressed for developing ontology editors and visualization tools. We mention the most commonly used tools as follows:

- *OntoEdit* [SEA+02], which is an ontology editor integrating numerous aspects of ontology engineering. It supports methodology-based ontology development with capabilities for collaboration and inference.

- *Ontolingua* [FFR97], which is an ontology tool created the Knowledge System Laboratory at Stanford University. Ontolingua consists of a set of tools and services to support the process of achieving consensus on common shared ontologies by geographically distributed groups. These tools make use of web technologies to enable wide access and provide users with the functionalities to publish, browse, create, and edit ontologies stored in an ontology server.

- *OILEd* [BHGS01], which is an ontology editor that has an easy to use frame interface, yet at the same time allows users to exploit the full power of an expressive web ontology language (OIL). OilEd uses reasoning to support ontology design, facilitating the development of ontologies that are both detailed and accurate.

- *UBOT* [KCH+02], which is a UML-based Ontology Tool set (UBOT) for building ontology engineering and natural language processing-based text annotation tools for DARPA Agent Markup Language (DAML). The UBOT tool extends UML by defining a prototype UML profile for DAML. This UML profile maps UML stereotypes to DAML-specific elements and for them to be used in the development of DAML ontologies.

- *Protégé* [GMF+03], which is an open-source knowledge acquisition and ontology development tool, developed by Stanford University. It provides a intuitive graphic user interface to define ontologies, and also includes deductive

classifiers to validate that models are consistent and to infer new information based on the analysis of an ontology.

In this thesis, we choose Protégé ontology editor to development our ontologies in the ontological framework. We will show the ontology development phase in Chapter 8.

## 6.5 KREM architecture: the *Rules* component

In the KREM architecture, the *Rules* component consists of different kinds of logical rules that allow reasoning over the individuals of the ontologies in the *Knowledge* component. Normally, these logical rules are in the IF-THEN format, which indicates that *if* a set of conditions is satisfied, *then* new knowledge of the individuals is inferred.

### 6.5.1 The Semantic Web Rule Language (SWRL)

Amenable to the Semantic Web standards, the Semantic Web Rule Language (SWRL) is a widely used standard language, proposed by the W3C community. SWRL is based on a combination of its sublanguages OWL DL and OWL Lite with the Rule-Markup Language. A SWRL rule is in the form of implication between an antecedent (body) and consequent (head), which can be interpreted in a way that whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold [HPSB$^+$04]. In SWRL, a rule has the syntax: $Antecedent \rightarrow Consequent$, where both the antecedent (body) and consequent (head) contains zero or more atoms. Atoms in SWRL rules can be the form of $C(x)$, $P(x,y)$, where $C(x)$ is an OWL class, P is an OWL property, and $x,y$ are either variables, OWL individuals or OWL data values [HPSB$^+$04].

In this work, we use SWRL rules to predict the machinery failures in manufacturing processes. During the prediction process, to improve the performance of the system, we address the rule conflict, subsumption, and redundancy issues (introduced in Chapter 11). The reason we choose SWRL rules is two-fold. Firstly, SWRL provides model-theoretic semantics and has the advantage of its close association with OWL ontologies, which enables the definition of complex rules for reasoning about individuals in ontologies. Secondly, the use of SWRL to write rules is inde-

pendent of rule implementation languages within rule engines, which has the advantage of the flexible selection of rule engines and inference platform.

To enable failure prediction, we use ontologies as well as SWRL rules to propose predictive rules. We will introduce our proposed approach in Chapter 8 (developed ontologies) and Chapter 9 (SWRL-based predictive rules).

### 6.5.2 Rule engines

A semantic reasoner, reasoning engine, rules engine, or simply a reasoner, is a piece of software able to infer logical consequences from a set of asserted facts or axioms [Abb12]. Rule engines provide mechanism to perform ontology reasoning, within which the inference rules are commonly specified by means of an ontology language (normally a description logic language).

In this subsection, we review the most recognized rule engines in the Semantic Web community, they are:

- *Jess*, the rule engine for Java platform [FH$^+$08]. It consists of a rule base, a working memory, an inference engine and an execution engine. Jess uses the Rete algorithm [WBG08] to match the declared facts in the working memory against the rules in the rule base. After the matching, system determines which of the rules should fire based on the facts. Within the rule engine, rules are written in the format of the Jess rule language or XML.

- *Jena inference engine*, a Java-based open-source application framework for developing Semantic Web applications adopted by the Apache Software Foundation [Jen15]. It is one of the development tools of the Jena framework. The Jena framework provides a set of pre-define reasoners to support user-defined rules in the Jena own syntax, and it support three execution strategies for rule reasoners: forward-chaining, tabled backward-chaining, and hybrid. The framework also support additional reasoning features such as preprocessing attachment, proof tracing, and proof explanation [Jen13].

- *JBoss Drools*, a rule engine that uses rule-based approach to implement Production Rule Systems [Bro09]. JBoss Drools implement and extends the Rete algorithm for object oriented systems. Within the rule engine, rules are stored in the production memory and the facts that the inference engine matches

against are kept in the working memory. To avoid conflict among a large number of rules, Drools implement the Agenda mechanism to manage the execution order of the conflicting rules, by using a Conflict Resolution strategy.

- *OpenRules*, an open source rule engine and a Business Decision Management System (BDMS) to represent and maintain business logic in the form of Executable Decision Models [DBRA+11]. It provide various tools for developing rule-based business decision support systems. Users can import and edit business rules based on Microsoft Excel, Microsoft Word, OpenOffice or Google Docs documents. This mechanism allows non-technical people to use a guided text editor for drafting business rules from decisoin tables. Also, a Eclipse plug-in is designed for technicians to edit rules in Java programming language.

## 6.6 KREM architecture: the *Experience* component

The distinctive advantage of the KREM architecture lies in its incorporation of the *Experience* component, which allows the capitalization and reuse of prior knowledge. Indeed, the knowledge models constructed in the *Knowledge* component can be incomplete [ZM15]. When the knowledge models suffer from the incompleteness issue, the *Experience* component performs as a complementary model to deal with the incomplete knowledge models. To do this, the *Experience* component collects the experience that is learned during the use of the knowledge-based system. In this way, the *Experience* component improves the knowledge model by progressively completing it with the experience acquired from the interventions of human experts. This process is termed as *experience capitalization*. By complementing the traditional architecture with the *Experience* component, KREM improves the efficiency of classic knowledge-based systems [ZM15].

There are two main techniques used in the literature for experience capitalization: case-based reasoning [Kol14] and Set of Experience Knowledge Structure (SOEKS) [SS09].

### 6.6.1 Case-based reasoning

Case-based reasoning (CBR) is a kind of analogical reasoning that focuses on reasoning based on previous experience [Kol14]. A piece of experience can play dif-

ferent kinds of roles in solving a specific problem, such as predict/warn a problem that will appear in the future, suggest a solution for solving a problem, or predict the potential effects of implementing a solution.

The main idea of CBR is to adapt the solutions that are used to solve old problems/cases for solving new problems/cases. During the problem-solving process, CBR enables the capitalization of experience in two phases: the experience collection phase and the experience reuse phase. To solve a new problem, CBR aims to find the old problems that are similar to the new one and construct a new solution either by reusing old solutions or by adapting them. If the old solutions are adapted for the new problem, they are stored in the case library for further use.

Normally, the cycle of CBR includes four steps [AP94]:

1) *Retrieve* the most similar case or cases to the new problem. When a new problem comes, it is standardized by case representation. After that, one or several similar case(s) are retrieved from the case library.

2) *Reuse* the information and knowledge in that case. In this step, a solution of a similar old problem is suggested for solving the new problem.

3) *Revise* the proposed solution. If the old solution can not be directly used for the new problem, it is revised according to previous experience, domain knowledge, and the context of the new problem.

4) *Retain* the parts of this experience likely to be useful for future problem-solving. At last, the useful part (comprises the new problem and its solution) of the new case is stored in the case library.

Case-based reasoning offers tremendous advantages over other AI-based techniques in all those fields where experiential knowledge is readily available [MJ+10]. It has been widely used a many domains, such as business [CP05], industry [BAB+03], medicine [HBSP05], and civil engineering [MRH02].

### 6.6.2 Set of Experience Knowledge Structure (SOEKS)

As another key technique for experience capitalization, SOEKS is a knowledge structure that combines organized information obtained from a formal decision event [SS09]. It applies the Deoxyribonucleic Acid (DNA) structure and human brain

mechanisms to store day-to-day experience that is collected from decision making events.

SOEKS is structured into four basic components: *Variables*, *Functions*, *Constraints*, and *Rules*. We introduce these four components as follows [SS09]:

1) *Variables* are the source of the other components and are the center root or the starting point of the structure. They are associated with cause and effect values that reflect the current and desired states of a system. A set of *Variables* usually involves representing knowledge with an attribute-value language.

2) *Functions* create links between dependent and non-dependent *Variables* for constructing multi-objective goals. These links are created based on the relationships and associations among the *Variables*.

3) *Constraints* are a special kind of *Functions*. They are connected to variables to specify their limits and boundaries. A set of *Constraints* also provide feasible solutions for solving a problem.

4) *Rules* are conditional relationships that operate in the universe of *Variables*. Normally, one *Rule* is a relationship that links a condition and a consequent with logic statements.

SOEKS facilitates knowledge retrieval, representation, and acquisition in decision-making processes. This structure has been used in a wide range of industrial applications, such as industrial maintenance systems [STV+07], virtual engineering systems [TGP+09], geothermal systems [SMASC09], embedded systems [ZSS10], and interactive TV [ZSS12], etc.

## 6.7 KREM architecture: the *Meta-Knowledge* component

The *Meta-Knowledge* component includes knowledge about the other three components. Meta-knowledge is the knowledge about domain knowledge, about rules, or about experience. Meta-knowledge can be either about the form of the representation scheme itself (e.g., its syntax), or about the knowledge that is represented in a knowledge base [B+79]. It can take the form of context, culture, or protocols to use this knowledge.

According to different contexts, the implementation of meta-knowledge may vary from each other. For example, in the medical domain, meta-knowledge can be the clinical terminologies that cover complex concepts such as diseases, operations, treatments, and medicines. When the symptoms and suspected disease of a patient change, the medical protocols, and terminologies used by physicians may also change. In the predictive maintenance domain, meta-knowledge can control the firing of different sets of rules according to different contexts. For example, the rules for identifying the dangerous threshold of machine temperature may change according to the humidity of the working environment of the machine. As another example, when a machine ages, the rules used for machinery failure prediction are different from the rules used at the beginning time of machine operation. Thus the rule base should be able to update itself concerning different environments. In this way, the use of Meta-knowledge can steer the execution of knowledge-based systems [ZMS19].

## 6.8 Summary

This chapter gives the theoretical foundations of knowledge-based systems. It starts with an introduction to the classic architecture of knowledge-based systems. After that, we introduced the KREM architecture, which is a novel and generic knowledge-based framework for problem-solving in engineering disciplines. At last, we gave a detailed demonstration of the four basic components of the KREM architecture. The demonstration includes the key methodologies, techniques, and real-world applications associated with each KREM component.

In the next chapter, we review the existing knowledge-based predictive maintenance systems. We pay special attention to those systems that use Semantic Technologies to facilitate predictive maintenance tasks in industry.

# Chapter 7

# Knowledge-based predictive maintenance systems: state of the art

## Contents

## 7.1 Introduction

The development of a knowledge-based predictive maintenance system requires domain knowledge about manufacturing processes and predictive maintenance to be represented in a formal way, thus making this knowledge usable by the system. To achieve this goal, semantic technologies, especially ontologies with their rule-based extensions, have shown promising capabilities for formalizing knowledge about predictive tasks in various domains [PC09, SWWP10]. By providing shared, rigorous, and machine-understandable vocabularies with robust structures, ontologies with their rule-based extensions enhance the semantic interoperability among different system components and system users.

In this chapter, we give a comprehensive review of the existing knowledge-based predictive maintenance systems. We address special attention on the ontological models and their rule-based extensions that are relevant to predictive maintenance. The review of the existing research works are categorized into three facets: i) ontological models and rule-based systems for modeling the manufacturing domain; ii) ontological models and rule-based systems for modeling the condition monitoring domain; iii) ontological models and rule-based systems for modeling the context.

The review is carried out following the methodology proposed in [TDS03]. The adopted methodology provides a systematic approach to guiding the review of a particular subject. This approach consists of three stages: i) planning the review; ii) conducting the review; and iii) reporting and dissemination [TDS03]. We begin with conducting a scoping study to identify the relevance of the literature with the domain of predictive maintenance. The scoping study includes the identification of keywords, clarification of inclusion and exclusion rules and selection of databases. As a result, related works that are relevant to predictive maintenance are collected and synthesized.

To clarify the scope and coverage of the review, we adapt the inclusion and exclusion rules for resources introduced in [LBS13]. The modified strategy is as follows:

- Ontological models and their rule-based extensions that are concerned with manufacturing system design, simulation, monitoring, planning and maintenance should be included.

- Multiple types of ontological models should be considered. These models include, but are not limited to formal ontologies, meta-models and data models.

- If a particular ontology has extensions, the most generic version should be chosen for analysis.

After the extensive literature review, we present in the following subsections the state-of-the-art of the ontological models and their rule-based extensions that are related to predictive maintenance. We specify the scope of these systems with identifying which domain (manufacturing, condition monitoring, or context) they serve.

## 7.2 Ontological models and rule-based systems for modeling the manufacturing domain

In the manufacturing domain, products are produced by manufacturing systems that consists of several production lines. Martin [MD03] asserts that a typical manufacturing system can be characterized according to three notions: i) *Product*; ii) *Process*; and iii) *Resources*. These three notions are interrelated, for example, a *Process* (e.g.,casting, melting, milling, drilling) can utilize *Resources* (e.g., machines, raw materials, tools) in a production system, and a *Process* can also be used to define *Product qualification procedure* [MD03].

Fig. 7.1 illustrates the integration of the three notions. *Product*, *Process*, *Resources* are represented in ovals with their corresponding representative examples. The three notions are integrated inside a single-line rectangle, indicating that under the framework of a single manufacturing process, the representation of these three notions is static. The double-line rectangles represent dynamic objects. To optimize the dynamic production objectives (e.g., quality, precision, sustainability) under different environments, the selection of models, methods and tools need to be customized to generate appropriate and feasible solutions. The solutions include the suitable design of production processes and test procedures, the correct selection of equipment, and validation of the producibility of products [MD03]. This generation necessitates the consideration of product constraints, such as the structure and functions of *Products*. On the other hand, *Resources* can also influence the quality of *Products* and *Processes*. In this way, the three notions are mutually related and coherently integrated. Working with these three notions helps to identify the inclusion and exclusion strategy of the reviewed research works.

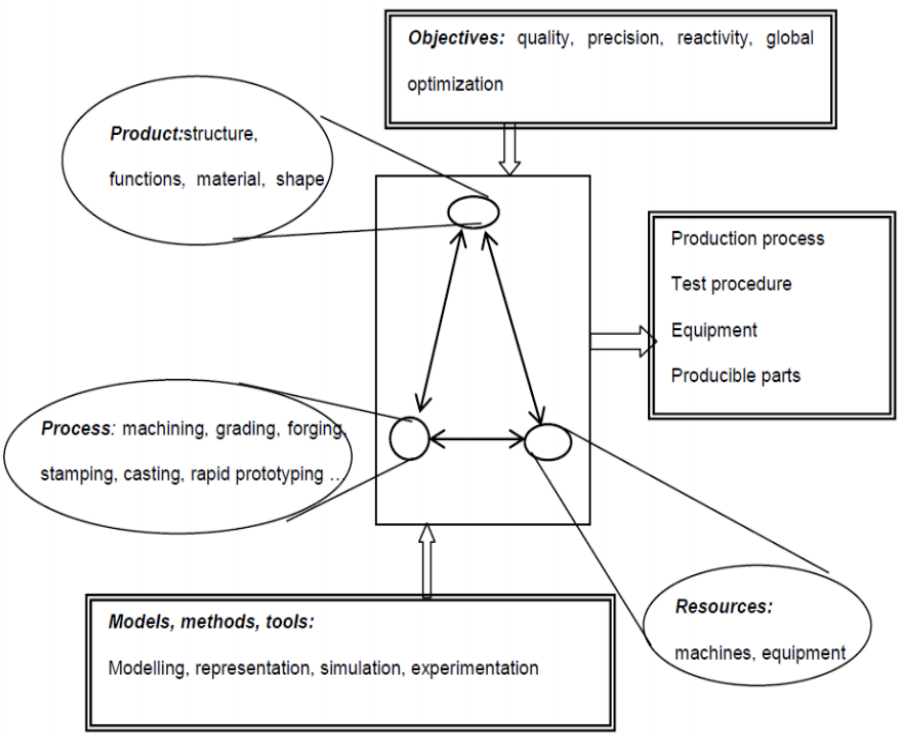



Figure 7.1 – Reference diagram of *Product/Process/Resource* integration [MD03].

### 7.2.1 Systems for modeling manufacturing products

During the physical product life cycle, the guarantee of information interoperability is of paramount importance. Typically, product information is stored and processed differently within distinct systems and enterprises. Inside these systems and enterprises, many participants and stakeholders have different domain engagement, domain knowledge and experience. This situation brings the issue of heterogeneity information to the domain, which may lead to the failure of the application system for achieving their intended design goals [CZMR18a]. This issue is a well-recognized problem that has been studied in the research field of interoperability problems. In this regard, many contributions about proposing common conceptual models for shared product information have been made.

To deal with the issues discussed previously, Vegetti et al. [VHL05] makes one significant contribution to developing the PRoduct ONTOlogy (PRONTO). Even though Product Data Management (PDM) Systems enable different organizations to share product data within the common product model, the lack of semantics integration has become a critical issue for heterogeneous systems. PRONTO is proposed

to overcome this issue. The main idea of PRONTO is to represent product-related concepts in different levels of abstraction. This multilevel formal representation enables heterogeneous systems to perform product-related planning actions at different aggregation levels [VHL05]. The ontology also specifies product structures with two aspects: composition and decomposition. This specification has been adopted to deal with complex product structure analysis in the industry.

However, PRONTO is not capable of referring to the existing standards related to the modeling of product structure, processes and features [PDT12]. To overcome this weakness, Panetto et al. [PDT12] makes remarkable efforts in developing ONTO-PDM, which contains the development of a common reference-information model. ONTOPDM harmonizes the product-related knowledge and standards, and the harmonization has shown positive results in solving the interoperability problems among different enterprises and applications. ONTO-PDM has the advantage of incorporating IEC 62264 and ISO 10303 standards, thus facilitates the management of heterogeneous product information. However, when ontological extension works need to be addressed, the required workload is high. Since other standardization initiatives may be considered to enrich the ONTO-PDM model, the mapping between different standards may require considerable efforts.

Like PRONTO and ONTO-PDM, the Process Specification Language (PSL) Ontology [GM03] is another notable development work in the subject area of product information modelling. PSL is developed by National Institute of Standards 6 and Technology (NIST), and it specifies logic terms with an ontology, for describing processes. The PSL ontology covers several domains such as manufacturing, engineering and business processes. In this section we only focus on the manufacturing domain. Even though it mainly focuses on the conceptualization of a process, the ontology also provides a basis for the formal description of elements and entities that constitute a Process. The foundation of the PSL ontology is four primary and disjoint concepts: i) *Activities*; ii) *Activity Occurrences*; iii) *Time Points*; and iv) *Objects*. From the manufacturing product point of view, the notion *Product* could be deemed as a sub-concept under the core concept *Objects* in the ontology. The PSL ontology provides a robust semantic foundation for modelling manufacturing product information. Furthermore, as indicated by the name, the PSL ontology is a powerful approach for the representation of manufacturing processes.

Another remarkable ontology dealing with manufacturing product modeling is the Product Semantic Representation Language (PSRL). Patile et al. [PDS05] devel-

oped this ontology-based framework for enabling semantic interoperability across different application domains. The authors use mathematical logic along with the standards-based approach to propose the semantic equivalence matrix. This matrix has been approved to be reliable for determining semantic equivalences between application ontologies and PSRL. These efforts form the foundation of seamless communication between product development systems. However, when the PSRL ontology is used to translate semantics of product information from one application to another application, there may exist the problem of information loss. To solve this issue, techniques such as text mining could be jointly used with PSRL, to avoid the missing of interesting information.

Besides the representative ontologies showed aborve, there exist other ontologies that have been developed to enhance the performance of product information modeling in the manufacturing domain. These ontologies and ontological models are: Supply Chain Operations Reference (SCOR) ontology [LPNG13], PLM ontology [BEC$^+$10], Product Search Intent Representation Scheme (PSIRS) [KCP08], MSDL [AD08], MAnufacturing ' s Semantics Ontology (MASON) [LSD$^+$], the ontology of a product data and knowledge management semantic object model for PLM [MK10], Parts Library Concept Ontology [MAP18] and Manufacturing System Engineering (MSE) ontology model [LH07].

Among them, the PLM ontology, MASON and MSE ontology models are considered as pertinent. The goal of the PLM ontology is to facilitate the share of product information by ensuring the transparent information interoperability among systems and people during the entire product lifecycle. The MASON ontology aims at providing a common semantic network in the manufacturing domain, by conceptualizing three core concepts: *Entities*, *Operations*, and *Resources*. The product information is specified with the superclass *Entities* and with subclasses such as *Geometric Entities for Manufacturing*, *Raw Material*, and *Assembly Entity*. Because of this, the MASON ontology presents a suitable approach for modeling product information that is incorporated in manufacturing processes.

Like the MASON ontology, the MSE ontology also conceptualizes multiple manufacturing domain notions. This ontology increases the level of cooperation among engineering team members, and this advantage enhances the degree of information autonomy in manufacturing processes. Among these three ontologies, the PLM ontology provides a more comprehensive and specific view of product information modeling. Compared to other two ontologies, one significant advantage of PLM is

the incorporation of *Duration of Time* concept into PLM models, which elevates the synchronization of different information systems.

### 7.2.2 Systems for modeling manufacturing processes

A manufacturing process is a sequence of activities through which the raw materials are assembled, integrated and transferred into a final product. These raw materials are firstly transferred into required parts of the product, after that combined to form the finished goods.

One notable approach to the formal representation of manufacturing processes is the PSL ontology, proposed by Grüninger et al. [GM03]. We have mentioned this ontology in the previous section from the manufacturing product point of view. However, the primary objective of the PSL ontology is to formalize process information. The formalism work covers subject areas related to process modeling, such as scheduling, process planning, simulation, work flow and project management. The PSL ontology framework incorporates a set of theories written in First-order Logic (FOL) [Smu12], and the framework consists of two parts: i) PSL-Core ontology; and ii) a set of extensions. The PSL-Core ontology is built upon a set of *Foundational Theories*, which include set theory and situation calculus [SST+00]. It is a set of intuitive semantic primitives written in the basic language of PSL [GM03]. These semantic primitives show adequate competence for describing the basics of manufacturing processes.

The foundation of the PSL-Core ontology is four primary and disjoint concepts: i) *Activities*; ii) *Activity Occurrences*; iii) *Time Points*; and iv). *Objects*. Making use of the situation calculus' own primitive "action", the concept Activities is defined as a class or type of actions. The concept *Activity Occurrences* gives the place and time of the occurred activity. In this regard, the concept *Activity Occurrences* conducts another class *Time Points*, which is also a primary concept. At last, anything that is not a *Time Point* or an *Activity* is treated as an *Object*. In this way, these four concepts form the PSL-Core ontology. Besides the PSL-Core ontology, several extensions are developed to supplement it. These extensions could be classified into two types: *Definitional Extensions* and *Non-definitional Extensions*. In the *Definitional Extensions*, all new linguistic items could be completely defined based on the *Foundational Theories* and the PSL-Core ontology. *Non-definitional Extensions* contain one or more notions that could not be defined with regards to the *Founda-*

*tional Theories* and the PSL-Core ontology. In this way, the PSL ontology provides a robust semantic foundation for modeling manufacturing process information. Fig. 7.2 shows the framework of the ontology. Notations relevant to the PSL-Core ontology are expressed with rounded rectangles. The role of *Foundational Theories* is to give precise definitions of the four primary concepts. Ovals represent the two types of extensions that were developed to describe the information which is not included in the PSL-Core ontology.

Another example is the MASON ontology. Lemaignan et al. [LSD+] makes a significant contribution when proposing this ontology for modeling manufacturing processes. The MASON ontology formalizes the *Process* notion by introducing a class named *Operations*. The ontology uses this class and its subclasses to specify a set of manufacturing-related processes, including manufacturing operations (e.g., machining operations, control, assembly), logistic operations (e.g., maintenance, handling), human operations (e.g., scheduling, programming) and launching operations (e.g., machine preparing).

Besides the PSL ontology and the MASON ontology, there exist other ontologies developed for modeling manufacturing processes, such as MSDL ontology [AD08], manufacturing reference ontology (MRO) [UYC+13], MSE ontology model [LH07] and ADAptive holonic COntrol aRchitecture for distributed manufacturing systems (ADACOR) ontology [BL07]. These ontologies have been successfully used for dealing with process design, planning and scheduling issues in the manufacturing domain. Among all ontologies we have described in this section, the PSL ontology provides the most general conceptualization of processes in various domains. However, because of the lack of manufacturing domain knowledge, this ontology is not suitable to be directly used for describing manufacturing domain activities. Compared to other ontologies, the MSDL ontology is a significant model for representing manufacturing processes, as it captures more specific and rigorous manufacturing domain knowledge.

### 7.2.3 Systems for modeling manufacturing resources

There are different definitions of the *Resources* notion. After the extensive survey, under the framework of the *Process* notion, we define the *Resources* notion as *physical objects that can execute a range of operations during a manufacturing process*.

Among the existing works, Borgo and Leitão [BL07] develop an ontology to for-

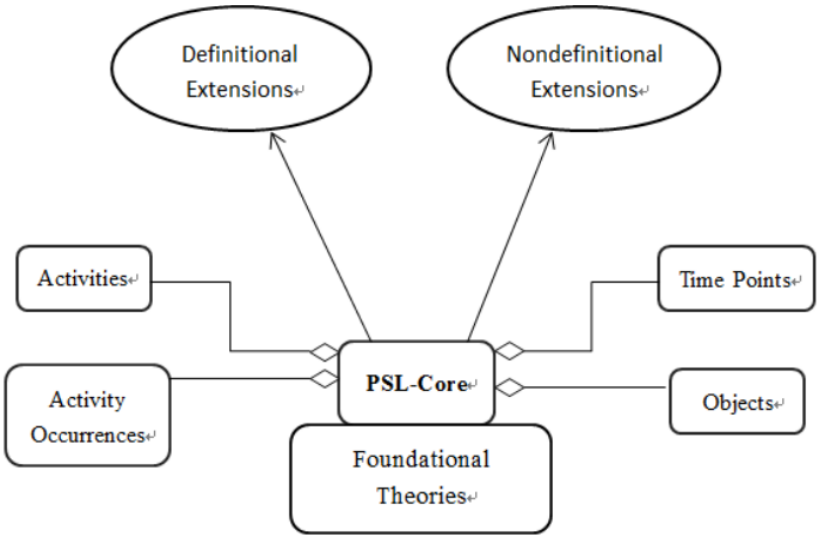



Figure 7.2 – The PSL framework.

mally describe manufacturing scheduling and control operations. They state that entities such as producer, transporter, tool are some examples of specializations of resources. In another influential ontology, Lin and Harding [LH07] build a link between the *Resources* and *Process* notions, by claiming that a *Process* uses *Resources*, and *Resources* can be efficiently allocated with considering the strategies and objectives of a *Process*.

Besides ontologies which merely give formal definition to the *Resources* notion, some ontologies also address the categorization task of the notion. In the MASON ontology, the *Resources* notion is classified into four sub-notions: i) *Machine-tools* (e.g., turning machines, drilling machines, milling machines); ii) *Tools* (e.g., forging die and punch, turning tool, founding pattern and mould); iii) *Human Resource* (e.g., procedure expert, handling operator, programming operator); and 4). *Geographical Resources* (e.g., plants, workshops). In a word, the *Resources* notion in the MASON ontology stands for the whole set of physical objects that are related to manufacturing [LSDS06].

In the MSDL ontology [AD06], the authors represent the *Resources* notion according to different levels of abstraction. The top-down representation is *Factory*, *Shop*, *Cell*, *Workstation* and *Machine*. The multiple-level abstraction of *Resources* eases the representation work of manufacturing services. To support intelligent supplier discovery, an SWRL rule-based extension of the MSDL ontology is developed in [AM14]. In their work, two categories of rules, namely, property infer-

ence rules and classification rules are introduced and implemented. Among them, the property inference rules are used for inferring new manufacturing capabilities based on the explicitly stated knowledge in the ontology. On the other hand, the classification rules are used for categorising suppliers under a certain class if they possess one or more required manufactiromg capabilities [AM14].

There are a set of ontological models that conceptualize the *Resources* notion with the aim of modeling domain knowledge in manufacturing. Other representative ontological models are the manufacturing core concepts ontology (MCCO) [UYC+11], ADACOR ontology [BL07] and MRO [UYC+13]. In Table 7.1, we summarize the representative ontologies and ontological models we have reviewed in this section, with regard to their scopes and typical applications.

Table 7.1 – Scopes and typical applications of the reviewd ontologies and ontological models in the manufacturing domain.

| Ontology name | Scope | Typical application |
|---|---|---|
| PRONTO | *Product* | Material requirements planning |
| ONTO-PDM | *Product, Process, Resources* | Simulation of distributed activities for manufacturing simple product prototypes |
| PSL | *Process, Resources* | Discrete-event simulation |
| MSDL | *Process, Resources* | Automation of various tasks throughout virtual enterprise life cycle |
| MASON | *Product, Process, Resources* | Multi-agent systems for manufacturing |
| MSE | *Product, Process, Resources* | Resource e-planning |
| MRO | *Product, Process, Resources* | Development of application-specific ontologies |
| ADACOR | *Product, Process, Resources* | Development of manufacturing control applications |
| MCCO | *Product, Process, Resources* | Explore interoperability across product lifecycle domains |

## 7.3 Ontologies and rule-based systems for modeling the condition monitoring domain

In order to schedule maintenance for avoiding production downtime, condition monitoring is a key technique to identify the functioning state of a machine or a mechanical system. It is an important task by which the machine or mechanical system deterioration tendency, and the location of failures can be detected. Generally aligned with fault diagnosis tasks, condition monitoring has appealed to deep

attention from domain experts and engineers. In the context of Industry 4.0, during the machine or mechanical system life cycle, condition monitoring activities are performed by obtaining data from sensors that are located at the components of the machines or mechanical systems, for identifying their operating state.

According to the current state of a machine, if any fault or failure exists, a diagnosis can be launched to determine the causes of it. Also, based on the characteristics of the fault or failure, analysis of how they will propagate and evolve over time can be performed. The use of condition monitoring techniques has several advantages, such as improved machine availability, improved production efficiency, and reduced maintenance cost [WY07, Rao96].

In recent years, ontologies and rule-based systems have attracted notable attention to enhance knowledge sharing in condition monitoring tasks, while offering a logically defined and controlled vocabulary of domain entities. This type of ontologies and rule-based systems normally focus on the issues of fault or failure prognostics and machine health monitoring.

Among the existing research efforts in the condition monitoring domain, the ontology introduced in [EFJ+10] is one of the earliest contributions. In their work, a domain-specific ontology [Fen01] is developed to streamline the implementation of industrial condition monitoring and to standardize the exchange of condition monitoring data. During the implementation of the condition monitoring system, the developed ontology serves as a commonly accepted data and knowledge representation schema for diagnosis-oriented maintenance.

The OntoProg Ontology [NB17] addresses the failure prediction of machines in smart factories. The ontology is developed based on a set of international standards, and a classification for severity criteria, detection, diagnostics, and prognostics of failure modes is provided. The ontology standardizes the concepts that are necessary for tackling machinery failure analysis tasks. SWRL rules are used together with the ontology to address failure analysis in mechanical components. The OntoProg Ontology is evaluated against the maintenance of a centrifugal pump. In the evaluation phase, SPARQL queries are used for verifying the fidelity, completeness, level of detail, robustness, and internal consistency of the ontology [NB17].

As another most recent contribution, the Sensing System Ontology [MBB+18] is developed to define the embedded sensing systems for industrial Product-Service Systems (PSSs). This ontology is used as the backbone of the PSS knowledge-based framework, and it describes the sensors that are embedded in PSSs for the aim of

providing customized services for users. The ontology is tested on an industrial use case whose objective is to perform health monitoring on laser cutting machines. In this use case, the Sensing System Ontology is used to support the knowledge repository of a collaborative condition monitoring system named ICP4Life platform [MBB+18].

The knowledge model for fleet condition monitoring, introduced in [MAVM11], is developed to handle contextual knowledge within a fleet scale. In their work, the authors develop a domain ontology to categorize fleet elements into three levels. This categorization of fleet elements enables the analysis of contextual data from different levels, thus enabling the analysis of abnormal health conditions from different components within a fleet system.

Knowledge-based systems are also used for condition monitoring in the wind energy domain. In [PC09], an ontology is developed and used as a basis of fault detection and diagnosis system for wind turbines' condition monitoring. The objective of the ontology is to model vital characteristics of a Wind Energy Converter's (WEC's) gearbox. It has been used together with SWRL rules and ontology queries to detect possible failures and their exact positions inside the WEC's gearbox.

In [ZYZ15], an intelligent fault diagnosis method is proposed based on ontologies and the Failure Mode, Effects, and Criticality Analysis (FMECA). Within the method, authors use ontologies and SWRL rules to reason about failure causes, the locations, and the diagnosis methods during the fault diagnosis process. The FMECA method is tested on a 1500 Series wind power turbine of a wind farm, where SWRL rules and JESS rule engine are jointly used to describe failure causality at different levels.

## 7.4 Systems for modeling the context

Due to the evolving nature of context-aware computing, computational entities in the industry are required to be context-aware so that they can adapt themselves to dynamically changing situations [WZG+04]. With the increasing need for formal context models, ontologies have contributed significantly in this field by facilitating context representation, context sharing and semantic interoperability of manufacturing systems. However, compared to the ontologies that model manufacturing and condition monitoring domains, ontologies for modeling the context are much less numerous.

Among the existing research works, the CONtext ONtology (CONON) [WZG+04] is one of the most recognized contributions. In their work, the authors summarize that *location*, *user*, *activity*, and *computational entity* are the most fundamental contextual elements for capturing the information about the executing situation of an entity. These four concepts form a set of upper-level concepts, thus providing flexible extensibility to add specific sub-concepts within different application domains [WZG+04]. To perform context reasoning, SWRL rules are proposed to check the consistency of context and deduce high-level context from low-level explicit context. The CONtext ONtology is validated on a smartphone scenario where a mobile phone can adapt its context according to different situations (location, activity, time) of users.

Similar to CONON, the formal context model developed in [GWPZ04] aims to address issues including semantic context representation, context reasoning and knowledge sharing, context classification, and context dependency. In this work, a wide range of contexts are classified into two main categories: direct context and indirect context. Direct context is acquired from a context provider (e.g., person, machine, system) directly. While indirect context is obtained by interpreting direct context through aggregation and reasoning process. Within this process, context reasoning is used to infer indirect context from other types of context. Based on the formal context model, a Service-Oriented Context-Aware Middleware (SOCAM) architecture is developed for building context-aware services [GWPZ04].

Another interesting work is the context ontology introduced in [PVdBW+04]. This adaptable and extensible context ontology is developed for creating context-aware computing infrastructures, ranging from small embedded devices to high-end service platforms. Using the developed ontology, the authors pay special attention on solve several key challenges in Ambient Intelligence, such as application adaptation, automatic code generation and code mobility, and generation of device specific user interfaces [PVdBW+04].

## 7.5 Summary

In this chapter, we have reviewed the existing ontologies and their rule-based extensions that are developed for knowledge-based predictive maintenance systems. The review is conducted with regard to three aspects: i) ontologies and rule-based systems for modeling the manufacturing domain; ii) ontologies and rule-based sys-

tems for modeling the condition monitoring domain; iii) ontologies and rule-based systems for modeling the context. We have demonstrated the scopes and objectives of different models, followed by introductions of typical applications of them.

We summarize the domain coverage of the existing knowledge models (ontologies and their rule-based extensions) in Table 7.2. We evaluate the domain coverage and scopes of these knowledge models by examining whether the key concepts required for describing the predictive maintenance domain are covered and formally described in these existing knowledge models. These key concepts can be categorized into three subdomains: *Manufacturing*, *Context*, and *Condition Monitoring*. For the *Manufacturing* subdomain, the key concepts are *Product*, *Process* and *Resource*. For the *Context* subdomain, the key concepts are *Identity*, *Activity*, *Time*, and *Location*. While for the *Condition Monitoring* subdomain, *Anomaly*, *Fault*, *Failure*, *Severity*, *Prognostics*, *Diagnostics*, *Alarm*, and *Alert* are the key concepts. These concepts form the columns of the Table 7.2, and the knowledge models are enumerated by rows. If a concept is covered by a knowledge model, a check mark is placed in the table. Otherwise, a cross mark is assigned.

After reviewing the knowledge models mentioned above, we recognize that none of them provides a satisfactory knowledge representation of the three subdomains. Some of these knowledge models focus on a narrow field, such as manufacturing resource planning, and they do not formalize predictive maintenance-related concepts, e.g., machinery *Failure* and *Fault*. Also, none of the existing knowledge models provide knowledge representation of the concepts related to *Warning Signal* in maintenance tasks, e.g., *Alert* and *alarm*. To perform a predictive maintenance task on a piece of machinery, the knowledge base of a knowledge-based system should incorporate not only the machine-interpretable knowledge for characterizing the manufacturing entities or processes which are being monitored but also the knowledge about fault or failure detection and prognostics. This motivates us to develop a more expressive and complete knowledge model that provides a rich representation of the domain knowledge in the fields of manufacturing, condition monitoring, and context.

Table 7.2 – A summary of the existing knowledge models with respect to their domain coverage.

| Ontologies | Context | | | | Condition Monitoring | | | | | | | Manufacturing | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Identity | Activity | Time | Location | Anomaly | Fault | Failure | Severity | Prognostics | Diagnostics | Alarm | Alert | Product | Process | Resource |
| MASON [LSD+] | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MSDL [AD08] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MRO [UYC+13] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| ONTO-PDM [VHL05] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MCCO [UYC+11] | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ |
| MaRCO [JSHL19] | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| MSE [LH07] | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ |
| ADACOR [BL07] | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| Wind turbine ontology [ZYZ15] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Sensing System Ontology [MBB+18] | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| OntoProg Ontology [NB17] | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ |

# Part III

# Contributions

# Chapter 8

# An ontological framework for knowledge-based predictive maintenance systems

## Contents

## 8.1 Introduction

As discussed in Chapter 6, the development of a knowledge-based predictive maintenance system requires domain knowledge about system operation and maintenance to be represented in a formal way, thus making this knowledge usable by the system. To achieve this goal, ontologies have shown promising results when formalizing knowledge about predictive maintenance tasks in various domains [PC09] [SWWP10]. However, as stated in Chapter 7, most of the existing ontologies and ontological models merely focus on a narrow field, while lacking the formal representation of all the knowledge required for predictive maintenance. This highlights the necessity for developing a novel ontological model that covers a broad range of concepts and relations that are necessary for describing the predictive maintenance domain.

In this chapter, an ontology-based framework that is used for the development of the knowledge-based predictive maintenance system is presented. The framework is proposed based on an ontological representation of condition monitoring knowledge in the manufacturing domain. The framework is presented by introducing an ontological structure which includes a core reference ontology for representing general condition monitoring concepts and relations, and a set of domain ontologies for formalizing manufacturing and condition monitoring domain-specific knowledge. The core reference ontology is aligned with the UFO ontology (Unified Foundational Ontology), which is an upper-level ontology providing general concepts and relations at a high abstraction level [GW10]. The domain ontologies specialize the core reference ontology into the manufacturing and condition monitoring domains, with representing domain-specific knowledge from different aspects such as context, product, process, and resources.

This chapter is structured as follows. Section 8.2 presents the ontological framework with its design methodologies. Section 8.3 shows the Condition Monitoring Core ontology (CM-core), which is a core reference ontology inside the framework that represents general condition monitoring knowledge. Section 8.4 introduces the Manufacturing Predictive Maintenance Ontology (MCMO), which is the key component of the ontological framework. Section 8.6 gives a case study on a conditional maintenance task of bearings in rotating machinery. In this case study, the ontological framework is jointly used with SWRL rules to perform rule-based reasoning. The rule-based reasoning results enables the identification of the conditions of these

bearings in rotating machinery. Section 8.7 introduces the evaluation of the developed ontologies according to their structure, function, and usability. Section 8.8 gives a summary of this chapter.

## 8.2 The proposed ontological framework

### 8.2.1 Ontology classification based on scope and domain granularity

The ontological framework consists of a set of ontologies with different abstraction levels. The abstraction levels of the ontologies are designed according to the ontology classification criteria introduced in [RPKC11], where ontologies are categorized into four layers according to their scope and domain granularity. Figure 8.1 shows the different layers of the ontology classification. From the top layer to the bottom, the scopes of ontologies become narrower, while the described concepts and relations in the ontologies become specific [CZMR18b].

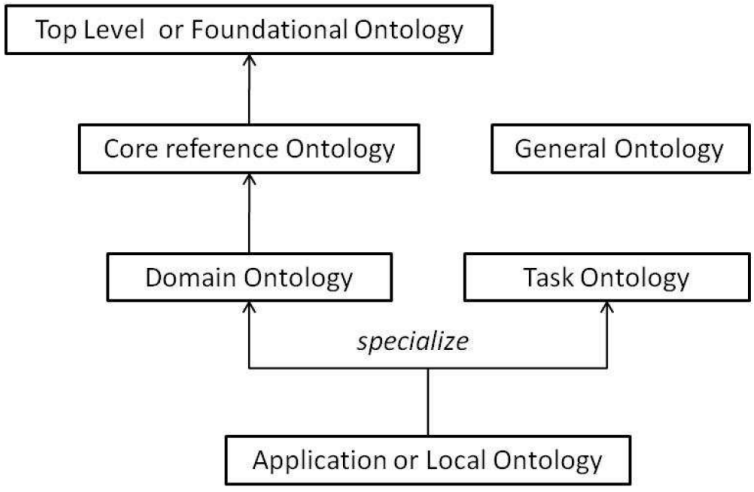



Figure 8.1 – The four-layer classification of ontologies [RPKC11].

In this work, the four-layer ontology classification is modified to propose a refined architecture. Within the refined architecture, only three layers of ontologies are considered: *Foundational ontology*, *Core (reference) ontology*, and *Domain ontology*. Since the objective is to represent the concepts and relations of predictive maintenance at a general level, *Application/Local ontology* in the four-layer structure is excluded. Fig. 8.2 shows the refined ontology architecture with three layers,

as well as the definitions of each layer of ontologies. From top to bottom, the specialization levels of ontologies become higher.

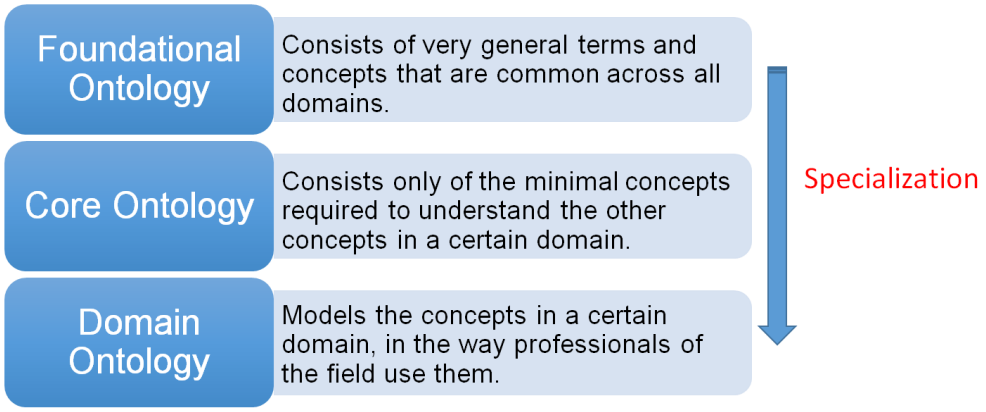



Figure 8.2 – The three-layer architecture of ontologies: *Foundational*, *Core*, and *Domain*.

To develop the ontological framework, the three-layer architecture is used as guidance for identifying the scopes and domain granularity of different ontologies. The following subsections show the development process and the key concepts/relations of these ontologies in detail.

### 8.2.2 Development methodology

For the development of the ontological framework, the iterative ontology development process introduced in [NM+01] is adopted. During this process, the following seven steps have been accomplished:

- Determine the domain and scope of the ontologies. In this step, the scopes of different ontologies in the framework are clarified, and the objectives and the domain coverage of the ontologies are determined. For example, the CM-core ontology demonstrates basic concepts that are common to a wide range of monitoring domains. While the MCMO ontology should provide specific domain concepts and relations. This step helps to clarify the abstraction levels of different ontologies.

- Consider the reuse of the existing ontologies and other information resources. The objective of this step is to determine which ontologies and resources are appropriate to be reused, refined or extended. For the ontological framework, the development work takes profit from a set of International Organization for

Standardization (ISO) standards such as ISO 13372 [ISO04], ISO 9000 [ISO05], ISO 9001 [SSGR09] and ISO 11783 [ISO07], as well as from the domain ontologies that are mentioned in Chapter 7.

- Enumerate all important terms in the ontologies. In the framework, the terms are extracted from three resources: i). ISO standards; ii). domain ontologies that are introduced in Chapter 7; and iii). relevant research papers and textbooks, such as [SSA+00] and [KM01].

- Define the classes and class hierarchy of the ontologies. This process includes the reuse of concepts in existing ontologies, and also the proposition of new concepts.

- Define the properties of classes. These properties give the internal structure of the concepts.

- Define the restrictions of the class properties. The class properties can have different restrictions on value types, allowed values, number of values and other features that the properties can take.

### 8.2.3 The three-layer framework

**Top layer: the foundational ontology**

The development of the ontological framework starts with the choice of a foundational ontology that defines general and basic notions across a wide range of domains. The reuse of a foundational ontology enables the integration of other ontologies that represents more specific domain concepts and relationships. In this thesis, the Unified Foundational Ontology (UFO) [GW10] is chosen, as it provides rigorous and expressive representation of general concepts and relationships. The UFO ontology adopts the *Endurant*/*Perdurant* dichotomy, in which an *Endurant* represents an entity comprising spatial components that is not dependent on any time frame of occurrence, while a *Perdurant* stands for an entity containing temporal components, and it presents only part of its temporal components at different time points.

The core ontology is then aligned to the UFO ontology, to ensure a rigorous conceptualization. The UFO ontology is at the top layer of our ontological framework.

### Middle layer: the core reference ontology for condition monitoring

After determining the foundational ontology, a core reference ontology for condition monitoring, named CM-core, is developed. According to [RPKC11], core reference ontologies are built within the scope of a domain. Normally, they catch central concepts and relationships of a domain and are considered as an integration of several domain ontologies [CZMR19]. During the development process, the Middle-Out approach for concept taxonomy construction [RPKC11] is used. The idea of this approach is to construct an ontology through the combination of the Top-Down and Bottom-Up approaches. To do this, the central concepts in the condition monitoring domain are identified. The central concepts are extracted from three sources: i). ISO standards 13372 [ISO12], 9000 [ISO05] and 9001 [ISO00]; ii). the domain ontologies relevant to condition monitoring; and iii). relevant research papers and textbooks, such as [SSA+00] and [KM01]. These central concepts are then generalized to the upper level, and the core reference ontology is aligned to the UFO ontology. On the other hand, these central concepts are specialized into different subdomains, for building domain ontologies. In this step, the upper-level concepts and relations are specialized into lower-level ones.

The CM-core ontology contains taxonomies of core condition monitoring concepts such as *system*, *function*, *behavior*, *structure*, *process*, *state*, *failure*, and *fault*, with their interrelationships. For defining these concepts, the existing ontologies such as the Semantic Sensor Network (SSN) Ontology [CBB+12], PSL Ontology [GM03], and SWRL Time Ontology [HP06] are resued.

### Bottom layer: domain ontologies for condition monitoring in manufacturing

A domain ontology represents specific domain knowledge and is only applicable to a certain domain. In this work, the CM-core ontology is specialized into domain ontologies using the Top-Down approach.

Among this level of ontologies, the *Manufacturing Condition Monitoring Ontology (MCMO)* plays the central role among all the domain ontologies. This ontology makes use of elements from other domain ontologies and represents knowledge from both manufacturing and condition monitoring domains. To enhance the reusability and extensibility of the MCMO ontology, the ontology partitioning and module extraction approaches introduced in [dSSS09] is adopted, and the ontology is structured into three modules. This ontology will be described in detail in Section

8.4.

The other domain ontologies are either reused or modified to propose the MCMO ontology. their main usage are as follows: the *Product ontology* aims to provide a comprehensive representation of manufacturing products and product components. The *Manufacturing Process ontology* gives a formal representation of manufacturing processes, such as cutting, drilling, milling, and casting. The *Manufacturing Resource ontology* provides knowledge about manufacturing resources which are physical objects used for executing a range of operations during different manufacturing processes. The *Context ontology* contains the representations of contextual knowledge, including the formal definitions of context entities such as person, activity, location and time. The *Sensor ontology* incorporates knowledge about sensors. This ontology is specialized from the core reference ontology SSN. A set of domain ontologies are reused in this step, including the MSDL ontology, PSL ontology, MASON ontology, .etc.

Fig. 8.3 shows the whole ontological framework, where the domain ontologies are presented at the bottom level. In the figure, ontologies are structured into the three-layered architecture. Rectangles with solid lines are different ontologies or conceptual models, and rectangles with dashed lines indicate different levels of domain granularity. Solid arrows indicate the alignment among ontologies. The alignment among ontologies is presented in the following subsections.

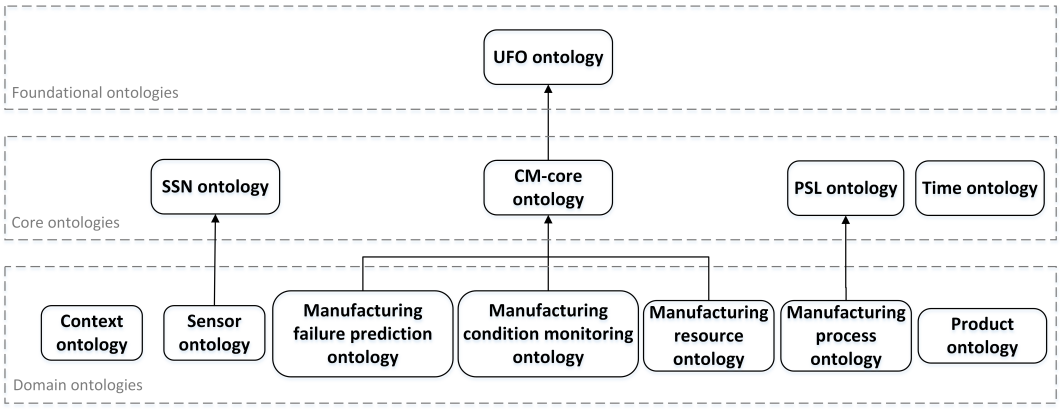



Figure 8.3 – The three-layer ontological framework.

## 8.3 The Condition Monitoring Core ontology (CM-core)

The CM-core ontology aims to cover general concepts in condition monitoring that are common to many subdomains. To permit generality, the formal representation of key condition monitoring entities in the CM-core ontology has been made as generic as possible. In this subsection, the terms that are associated with the classes of the ontology are presented. The global architecture of the ontology will be presented thereafter.

The terms are extracted from a set of ISO standards: ISO 13372 [ISO04], ISO 9000 [ISO05], ISO 9001 [SSGR09] and ISO 11783 [ISO07]. Also, relevant textbooks such as [SSA$^+$00] and [KM01] are reused. The definitions of these terms in the CM-core ontology are given as follows:

- *System*: a set of interrelated elements that achieve a given objective through the performance of a specified function [Ste16]. The definition of the class *OntoProg: System* in [NB18] is refined, by using a qualitative characterization model called Structure-Behavior-Function (SBF) [GRV09]. This class has been aligned with the *Object* class in UFO, through the alignment

$$\mathrm{CM}-core:\mathrm{S}ystem \sqsubseteq \mathrm{UFO}:\mathrm{O}bject.$$

  In UFO, *Object (Endurant)* represents a set of entities that posses spatial-temporal qualities. In CM-core, the *System* class contains different kinds of physical systems in the manufacturing domain, and they all obtain spatial-temporal qualities. Thus, the above alignment is proposed.

- *Process*: a set of interrelated or interacting activities that use inputs to deliver an intended result [SSGR09]. This class has been aligned with the *Complex Event* class in UFO, through the alignment

$$\mathrm{CM}-core:\mathrm{P}rocess \sqsubseteq \mathrm{UFO}:\mathrm{C}omplex\,\mathrm{E}vent.$$

  In UFO, a *Complex Event (Perdurant)* stands for a set of combined events with each consists of at least two single events. The *Process* class in CM-core repre-

sents manufacturing processes that are structured as a series of simple activities.

- *Parameter*: a variable representing some significant measurable system characteristics. The value of a *Parameter* is measured by sensors that are located at different components of a *System*. When the domain changes, a different set of *Parameters* could be selected for the condition monitoring task, such as bearing temperature, oil debris, oil pressure, velocity, voltage, etc. The alignment for this class and the upper-level UFO class is

$$\text{CM} - core : \text{P}arameter \sqsubseteq \text{UFO} : \text{M}oment.$$

  In UFO, the *Moment (Endurant)* class is a set of individuals that are existentially dependent on individuals of *Object*, such as color, height and electronic charge. In CM-core, a *Parameter* is also defined to be existentially dependent on a *System*.

- *State*: the condition of a *System* at a specific time. The condition of a *System* is associated with the values of a set of *Parameters*. Since a *State* is existentially dependent on a *System*, the alignment

$$\text{CM} - core : \text{S}tate \sqsubseteq \text{UFO} : \text{M}oment$$

  has been proposed.

- *Behavior*: it is represented as a sequence of *States* and transitions among them. The *States* together with transitions specify the evolution in the values of the *Parameters* [GRV09]. According to the definition, the *Behavior* of a *System* is also existentially dependent on a *System*. Therefore, the alignment

$$\text{CM} - core : \text{B}ehavior \sqsubseteq \text{UFO} : \text{M}oment$$

  has been introduced for the CM-core ontology.

- *Fault*: a *Fault* is defined as a condition of a *System* that occurs when one of its components or assemblies degrades or exhibits abnormally [ISO04]. The representation of this class is adapted from *OntoProg: Fault* [NB18], with slight

modification. Based on the definition, the alignment

$$\text{CM} - core : \text{F}ault \sqsubseteq \text{UFO} : \text{E}vent$$

is proposed, for representing a *Fault* as an *Event (Perdurant)* that happens in time.

- *Failure*: termination of the ability of an item to perform a required function [ISO04]. A *Failure* is a manifestation of a *Fault*. For the definition of this term, the definition of the class *OntoProg: Failure* [NB18] is adapted, with minor alteration. This class is also algined with the UFO class *Event*, through the alignment

$$\text{CM} - core : \text{F}ailure \sqsubseteq \text{UFO} : \text{E}vent.$$

Based on these terms, the classes with their object properties and data properties are constructed. A class defines a group of individuals which share some properties, and individuals are instances of the class. Object properties are binary relations between individuals, and data properties can relate individuals to concrete data values. Fig. 8.4 shows the global architecture of the CM-core ontology. For the purpose of clarity, only part of the classes and properties are shown. In Fig. 8.4, rectangles are classes, solid lines represent object properties between classes. The CM-core ontology is developed in OWL.

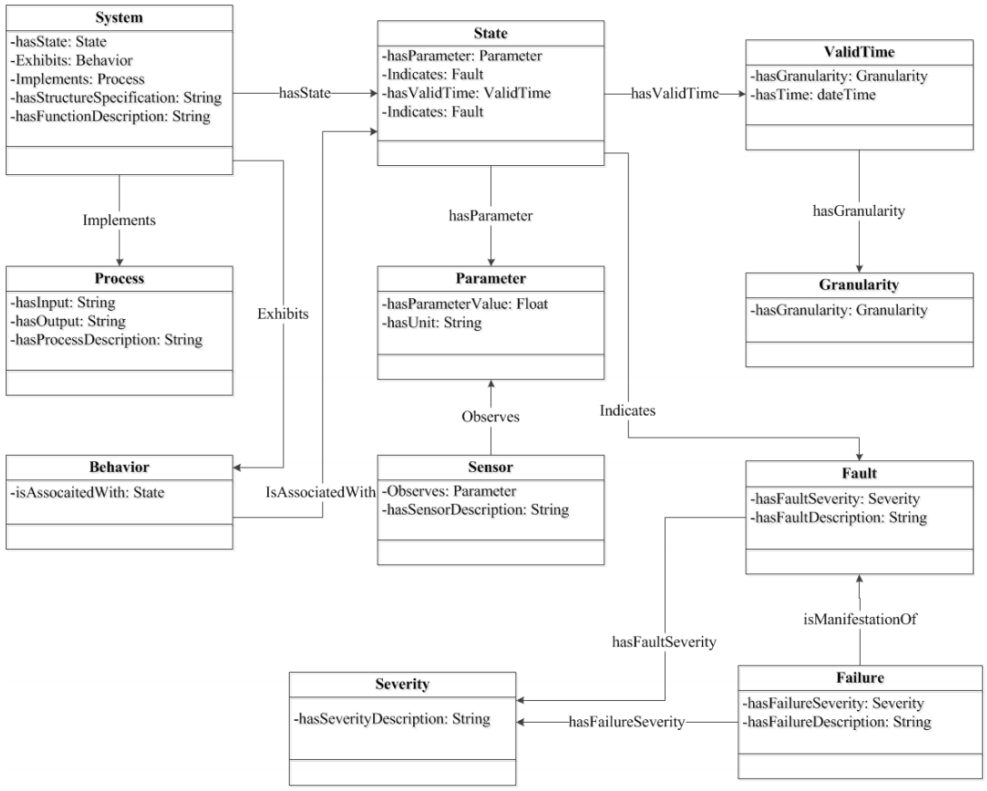



Figure 8.4 – The global architecture of the CM-core ontology.

## 8.4 The Manufacturing Condition Monitoring Ontology (MCMO)

At the bottom layer of the ontological framework shown in Fig. 8.3, a domain ontology named MCMO is developed. However, the MCMO ontology consists of a large number of domain concepts and relations, which hardens the management and maintenance of the knowledge [CGZM+19]. To cope with this issue, designing an ontology into separate knowledge components is an appropriate way to enhance its reusability and extensibility [dSSS09]. This design methodology is called ontology modularization, which stands for the method of structuring an ontology into different modules. The separate, independent, and reusable modules can facilitate the management and exploitation of the ontology. In this work, the ontology partitioning and module extraction approach introduced in [dSSS09] has been followed. As a result, the MCMO ontology is structured into three modules: the *Manufacturing Module*, the *Context Module*, and the *Condition Monitoring Module*.

The MCMO ontology consists of thirteen super-classes, assigned to three on-

tology modules. In this section, the three modules and the descriptions of the key classes are shown in detail, which enables the construction of the classes and the class hierarchy. The axioms defining the main classes of our ontology will be presented below using the description logic (DL) syntax [KSH12]. The compactness of this syntax is more suitable than its Web Ontology Language (OWL) counterpart for human reading. Nevertheless, all the ontologies presented in this thesis have been implemented in OWL.

### 8.4.1 The *Manufacturing Module*

The *Manufacturing Module* models the manufacturing domain according to three notions: *Product*, *Process*, and *Resource*. Among them, the *Product* notion formalizes the manufacturing product information during the product life cycle, the *Process* notion gives the conceptualization of a set of manufacturing processes, and the *Resource* notion describes the knowledge about the resources, which are used by production systems to produce manufacturing products. In our ontology, the classes used for describing the *Manufacturing* domain are as follows.

- *ManufacturingResource*: this class describes the resources that are used for performing manufacturing activities. It is further divided into three subclasses: *FinancialResource*, *HumanResource*, and *PhysicalResource*. Among them, *PhysicalResource* is a key class which stands for a set of physical entities that the condition monitoring tasks are performed upon, such as machines, machine tools, and workpieces. To define this class, the definition of the class *MASON: Resource* in MASON ontology [LSDS06] is extended. The DL axioms for defining this class and the *PhysicalResource* class are

$$\mathrm{M}anufacturing\mathrm{R}esource \equiv \mathrm{H}uman\mathrm{R}esource \sqcup \mathrm{P}hysical\mathrm{R}esource \sqcup \mathrm{F}inancial\mathrm{R}esource,$$

  and

$$\mathrm{P}hysical\mathrm{R}esource \sqsubseteq \mathrm{M}anufacturing\mathrm{R}esource \sqcap \exists has\mathrm{L}ocation.\mathrm{L}ocation \sqcap \exists isInsideOf.\mathrm{M}anufacturing\mathrm{F}acility \sqcap \forall \mathrm{E}xists\mathrm{I}n^{-1}.Fault.$$

- *ManufacturingProcess*: this class describes different types of manufacturing processes, which are structured sets of operations that transfer raw materi-

als or semi-finished product segments into further completed product parts. This definition is adapted from *MSDL: Process* [AD06], and the DL axioms for defining this class are

$$\mathrm{M}anufacturing\mathrm{P}rocess \equiv \mathrm{A}ssembly\mathrm{P}rocess \sqcup \mathrm{F}inishing\mathrm{P}rocess \sqcup \mathrm{F}orming\mathrm{P}roces \sqcup \mathrm{M}achining\mathrm{P}rocess \sqcup \mathrm{M}oulding\mathrm{P}rocess,$$

and

$$\mathrm{M}anufacturing\mathrm{P}rocess \equiv \exists \mathrm{M}akes\mathrm{U}se\mathrm{O}f.\mathrm{M}anufacturing\mathrm{R}esource \sqcap \exists has\mathrm{P}rocess\mathrm{I}nput.\mathrm{W}orkpiece \sqcap \exists has\mathrm{P}rocess\mathrm{O}utput.\mathrm{R}ealized\mathrm{P}art.$$

- *ManufacturingFacility*: this class consists of different places or areas that *ManufacturingProcesses* are executed. The definition of this class and its subclasses are adatped from *MRO: ManufacturingFacility* [UYC+13]. The DL axioms for defining this class are

  $$\mathrm{M}anufacturing\mathrm{F}acility \equiv \mathrm{S}tation \sqcup \mathrm{C}ell \sqcup \mathrm{S}hop \sqcap \mathrm{F}actory \sqcup \mathrm{E}nterprise,$$

  and

  $$\mathrm{M}anufacturing\mathrm{F}acility \sqsubseteq \exists has\mathrm{C}apability\mathrm{F}or.\mathrm{M}anufacturing\mathrm{P}rocess \sqcap \exists \mathrm{P}roduces.\mathrm{R}ealized\mathrm{P}art.$$

- *RealizedPart*: this class represents the physically existential parts of a product, which are obtained after the execution of one or several *ManufacturingProcesses*. This definition is adapted from the concept *MRO: RealisedPart* in [UYC+13], with slight alteration. The DL axiom used for defining this class is

  $$\mathrm{R}ealized\mathrm{P}art \sqsubseteq \exists has\mathrm{P}art\mathrm{S}tate.\mathrm{R}ealized\mathrm{P}art\mathrm{S}tate \sqcap \exists \mathrm{P}roduces^{-1}.\mathrm{M}anufacturing\mathrm{F}acility \sqcap \exists \mathrm{T}ransforms\mathrm{I}nto^{-1}.\mathrm{R}aw\mathrm{M}aterial \sqcap has\mathrm{P}rocess\mathrm{O}utput^{-1}.\mathrm{M}anufacturing\mathrm{P}rocess.$$

### 8.4.2 The *Context Module*

The *Context Module* formalizes the knowledge related to context. According to the definition in [BDR07], the required concepts for describing the context are: *Identity*

(a unique identifier for discriminating an entity), *Activity* (intrinsic characteristics of an entity), *Time* (timestamps for recording an entity's situation), and *Location* (an entity's position, co-location, or proximity) [DAS01]. In our ontology, classes used for describing the context are given below.

- *State*: this class describes the condition of a manufacturing entity or process, which is being monitored. A *State* is associated with a timestamp for providing temporal information, and with a set of *Parameters* which indicate the correctness of it. The DL axioms for defining this class are

$$State \equiv ProcessState \sqcup ResourceState \sqcup RealizedPartState,$$

and

$$State \sqsubseteq \forall hasParameter.Parameter \sqcap (\geq 1\ hasParameter.Parameter) \sqcap \exists hasValidTime.ValidTime \sqcap \forall Indicates.Error.$$

The definition of *ValidTime* and *Parameter* will be given later in this section.

- *ValidTime*: this class gives temporal information of a *State*. It has two subclasses, named *ValidInstant* and *ValidPeriod*. The description and definition of this class is adopted from *Temporal: ValidTime*, in [OD10]. To define this class, the following two DL axioms are introduced.

$$ValidTime \equiv ValidInstant \sqcup ValidPeriod,$$

and

$$ValidTime \sqsubseteq \forall hasGranularity.TimeGranularity \sqcap \forall hasValidTime^{-1}.State.$$

- *TimeGranularity*: this class specifies the granularity of a *ValidTime*. Individuals of this class are *Year*, *Month*, *Day*, *Hour*, *Minute*, *Second*, and *Millisecond*. The definition of this class is adapted from the concept *Temporal: Granularity* [OD10], and the DL axiom is

$$TimeGranularity \sqsubseteq \forall hasGranularity^{-1}.ValidTime.$$

- *Parameter*: variable which represents some significant measurable character-

istic of a monitored *ManufacturingProcess*, *PhysicalResource* or *RealizedPart*. The value of a *Parameter* is measured by sensors which are located at different components of the monitored entity. The DL axioms for defining this class are

$$\mathrm{P}arameter \equiv \mathrm{M}achine\mathrm{P}arameter \sqcup \mathrm{M}achinery\mathrm{P}art\mathrm{P}arameter \sqcup \mathrm{M}achine\mathrm{T}ool\mathrm{P}arameter \sqcup \mathrm{P}rocess\mathrm{P}arameter \sqcup \mathrm{R}ealized\mathrm{P}art\mathrm{P}arameter,$$

and

$$\mathrm{P}arameter \sqsubseteq \exists has\mathrm{P}arameter^{-1}.\mathrm{S}tate.$$

- *Location*: this class allows to specify the spatial information of a *PhysicalResource*. The definition of this class is adapted from the concept *Context: Location* in [BDR07]. Based on the definition, the following DL axiom is proposed.

$$\mathrm{L}ocation \sqsubseteq \exists has\mathrm{L}ocation^{-1}.\mathrm{P}hysical\mathrm{R}esource.$$

### 8.4.3 The *Condition Monitoring* Module

Condition monitoring is a broad research topic. In this subsection, we focus on the issue of prognostics and health management (PHM) in manufacturing. In [NB17], the authors pointed out that under the context of PHM, condition monitoring is the best way to minimize the probability of machinery failure occurrence and maintenance cost. In this context, the *Condition Monitoring Module* in our ontology aims to represent knowledge that is essential for describing condition monitoring tasks. The classes under this ontology module are

- *Fault*: this class contains different types of *Faults* that may exist in *PhysicalResources*. A *Fault* is a condition of a *PhysicalResource* under which the components of the *PhysicalResource* degrades from the theoretically correct condition. A *Fault* is considered to be *active* if it produces an *Error*, otherwise, it is *dormant* [ALRL04]. This characteristic of *Fault* is described by defining the DL axioms as follows.

$$\mathrm{F}ault \equiv \mathrm{A}ctive\mathrm{F}ault \sqcup \mathrm{D}ormant\mathrm{F}ault,$$

$$\mathrm{F}ault \sqsubseteq \forall \mathrm{E}xists\mathrm{I}n.\mathrm{P}hysical\mathrm{R}esource,$$

and

$$\mathrm{A}ctive\mathrm{F}ault \sqsubseteq \exists \mathrm{A}ctivates.\mathrm{E}rror.$$

- *Error*: this class specifies different types of *Errors* that could be activated by *Faults*. An *Error* is the discrepancy between a measured value of a *Parameter* and the theoretically correct value [ALRL04]. The DL axiom for defining the *Error* class is as follows.

$$\mathrm{E}rror \sqsubseteq \forall \mathrm{P}ropagates\mathrm{I}nto.\mathrm{F}ailure \sqcap \forall \mathrm{T}riggers.\mathrm{W}arning\mathrm{S}ignal \sqcap \exists \mathrm{A}ctivates^{-1}.\mathrm{A}ctive\mathrm{F}ault \sqcap \forall \mathrm{I}ndicates^{-1}.\mathrm{S}tate.$$

  The description of the *Failure* and *WarningSignal* classes will be demonstrated later on.

- *Failure*: this class represents the *Failures* that may occur in *PhysicalResources*. A *Failure* is the inability of an entity to perform one required function, and it can be the result of a propagation of an *Error* [NB17]. The axiom for defining this class is

$$\mathrm{F}ailure \sqsubseteq \forall \mathrm{P}ropagates\mathrm{I}nto^{-1}.\mathrm{E}rror.$$

- *WarningSignal*: this class stands for the auditory or visual signals for revealing the existence of one or several *Errors*, which indicate the reduction of the performance of *Physical Resources*, if unattended. *Alarm* and *Alert* are two typical types of *WarningSignals* in the manufacturing domain, between which *Alarm* warns about a higher severity level problem than *Alert*. Based on the definition, the following two axioms are proposed.

$$\mathrm{W}arning\mathrm{S}ignal \equiv \mathrm{A}larmt \sqcup \mathrm{A}lert,$$

  and

$$\mathrm{W}arning\mathrm{S}ignal \sqsubseteq \exists \mathrm{T}riggers^{-1}.\mathrm{E}rror.$$

Fig. 8.5 gives the global architecture of the proposed ontology, in which the round rectangles denote classes, solid lines stand for *is-a* or *subsumption* relationships, and dashed lines represent object properties. The classes are sorted into three categories, among which the classes with grey background belong to the *Manufacturing Module*, classes with black background pertain to the *Context Module*, and

classes with white background belong to the *Condition Monitoring Module*.

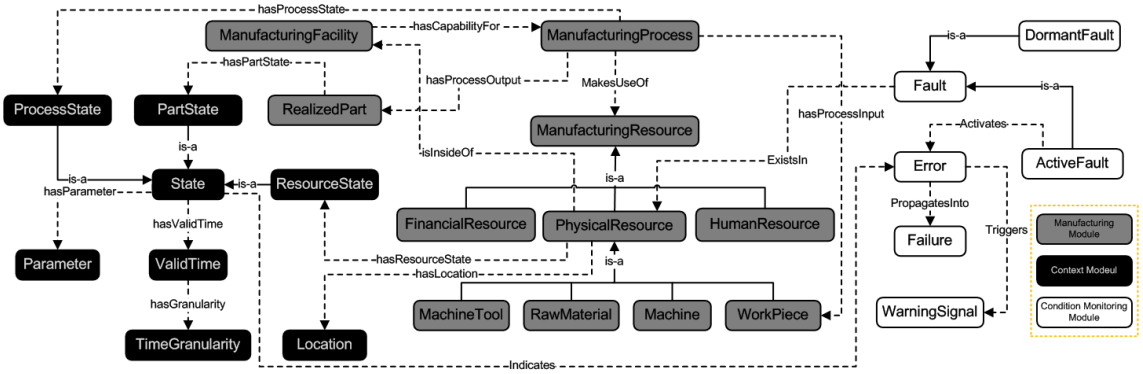



Figure 8.5 – Global architecture of the MCMO ontology.

## 8.5 An extension of MCMO: The Machinery Failure Prediction Ontology (MFPO)

The MCMO ontology models domain knowledge about condition monitoring at a general level. When the scope is limited in failure prediction, MCMO is not capable of providing specific knowledge about different types of events (non-failure events, failure events) as well as their temporal information. This motivates us to develop a more specific domain ontology named Machinery Failure Prediction Ontology (MFPO), which extends the MCMO ontology and focus on the modelling of essential knowledge for failure prediction. The MFPO ontology provides the foundation to formally represent events with their time information, for the purpose of failure prediction.

To describe the main classes in the MFPO ontology, UML notation is used [KCH+02]. The UML diagram for describing the main classes is shown in Fig. 8.6, where the boxes stand for ontology classes, and the arrows represent object properties. Data properties are indicated by class attributes. For the purpose of clarity, only a subset of the whole classes and relationships are presented.

After demonstrating the UML diagram of the MFPO ontology, the axioms of the main classes are introduced. Similar to the axioms describing the MCMO ontology, the axioms defining the main classes are presented below using the description logic (DL) syntax [KSH12].

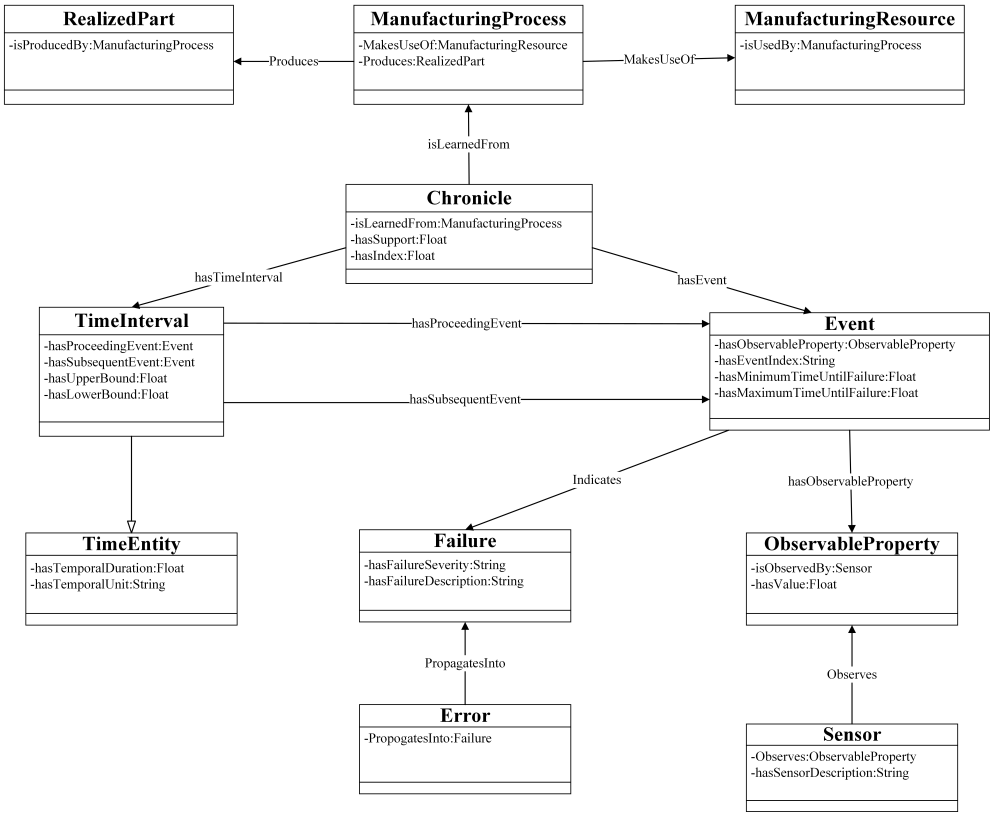



Figure 8.6 – The main classes in the MFPO ontology.

- *ManufacturingResource*: This class describes the resources that are used within manufacturing processes. It consists three subclasses: *FinancialResource*, *HumanResource*, and *PhysicalResource*. Among the three subclasses, *PhysicalResource* stands for a set of physical entities that the predictive maintenance tasks are performed upon, such as machine tools, workpieces, and final products. The definition of this class is extended from the class *MASON: Resource*, in the MASON ontology [LSDS06]. The DL axioms for defining this class and the *PhysicalResource* class are

$$\mathrm{M}anufacturing\mathrm{R}esource \equiv \mathrm{H}uman\mathrm{R}esource \sqcup \mathrm{P}hysical\mathrm{R}esource \sqcup \mathrm{F}inancial\mathrm{R}esource,$$

and

$$\mathrm{M}anufacturing\mathrm{R}esource \sqsubseteq \forall \mathrm{M}akes\mathrm{U}se\mathrm{O}f^{-1}.\mathrm{M}anufacturing\mathrm{P}rocess.$$

- *ManufacturingProcess*: It describes different types of structured sets of operations that transform raw materials or semi-finished product segments into further completed product parts [CGZM+19]. The DL axioms for defining this class are

$$\mathrm{M}anufacturing\mathrm{P}rocess \equiv \mathrm{A}ssembly\mathrm{P}rocess \sqcup \mathrm{F}inishing\mathrm{P}rocess \sqcup \mathrm{F}orming\mathrm{P}roces \sqcup \mathrm{M}achining\mathrm{P}rocess \sqcup \mathrm{M}oulding\mathrm{P}rocess,$$

and

$$\mathrm{M}anufacturing\mathrm{P}rocess \sqsubseteq \exists \mathrm{M}akes\mathrm{U}se\mathrm{O}f.\mathrm{M}anufacturing\mathrm{R}esource \sqcap \exists has\mathrm{P}rocess\mathrm{I}nput.\mathrm{W}orkpiece \sqcap \exists \mathrm{P}roduces.\mathrm{R}ealized\mathrm{P}art.$$

- *Chronicle*: Chronicles are a special type of sequential patterns, in which temporal orders of events are quantified with numerical bounds [SMS+19]. To

introduce this concept in the MFPO ontology, the following axiom is used.

$$\begin{aligned} &Chronicle \sqsubseteq \forall has\text{Event}.\text{Event} \sqcap (\geq 1 has\text{Event}.\text{Event}) \\ &\sqcap \forall has\text{TimeInterval}.\text{TimeInterval} \sqcap \\ &(\geq 1\ has\text{TimeInterval}.\text{TimeInterval}) \sqcap \\ &\exists is\text{LearnedFrom}.\text{ManufacturingProcess}. \end{aligned}$$

- *Event*: In predictive maintenance tasks, an *Event* is generally associated with a set of *ObservedProperties* which indicate the correctness of the operation of a piece of machinery. In this context, the DL axioms for defining this class is

$$\begin{aligned} &\text{Event} \equiv \forall has\text{ObservedProperty}.\text{ObservedProperty} \sqcap \\ &(\geq 1\ has\text{Observed}.\text{Property}). \end{aligned}$$

- *ObservedProperty*: This is an attribute which represents some significant measurable characteristic of a monitored *ManufacturingProcess*, *ManufacturingResource* or *RealizedPart*. The value of an *ObservedProperty* is measured by sensors which are located at different components of the monitored entity. This class is also called *Attribute*. The DL axioms for defining this class are

$$\begin{aligned} &\text{ObservedProperty} \sqsubseteq \exists has\text{ObservedProperty}^{-1}.\text{Event} \sqcap \\ &\exists \text{Observes}^{-1}.\text{Sensor}. \end{aligned}$$

- *Failure*: This class represents the *Failures* that are indicated by *Events*. A *Failure* is the inability of an entity to perform one required function, and it can be the result of a propagation of a machinery error [ALRL04]. The following axiom is used to define this class:

$$Failure \sqsubseteq \forall \text{Propagates}Into^{-1}.\text{Error}.$$

- *TimeInterval*: A temporal entity with an extent or duration. The definition of this class is adopted from the Time Ontology [HP06]. The axiom for describing this class is

$$\begin{aligned} &TemporalInterval \sqsubseteq \exists has\text{Proceeding}\text{Event}.\text{Event} \sqcap \\ &\exists has\text{Subsequent}\text{Event}.\text{Event} \sqcap \exists has\text{TimeInterval}^{-1}.\text{Chronicle}. \end{aligned}$$

By providing formal logical axioms and rich representations of machine-interpretable semantics for databases, the MFPO ontology can support semantic interoperability among different systems and system components. Also, the MFPO ontology provides rich representations of machine-interpretable semantics for knowledge-based predictive maintenance systems. This ensures the predictive maintenance systems can interoperate with shared semantics and a high level of semantic precision.

## 8.6 Rule-based reasoning for the condition monitoring of rotating machinery: a case study

To enable ontology reasoning for the goal of machine defects prediction, SWRL rules are used to perform reasoning tasks on individuals, through which new knowledge about these individuals can be inferred. The SWRL rules are extracted from a real-world experiment about bearing defects identification, introduced in [BDRC10].

With the goal of validation, we instantiate the MCMO ontology with different examples of bearings in rotating machinery, during which the ontology provided rigorous and formalized knowledge representation for modeling the condition monitoring tasks of these bearings. The proposed SWRL rules inferred about the correctness of the bearing operating states, thus enabling further maintenance actions to be scheduled.

During the operation stage of rotating machinery, vibration is a significant indicator chosen for detecting the partial or complete degradation of machinery components. In the literature [BDRC10], the authors summarize that the most relevant indicators for detecting bearing defects are *Kurtosis*, *Crest Factor*, and the *Root Mean Square (RMS)* value of the vibratory signal. Based on the analysis of the numeric values of these three parameters, the bearing states can be categorized into four classes, for representing their identities: Bearing without defect, Bearing with inner race defect (Dir), Bearing with outer race defect (Dor), and Bearing with two defects (Dor_ir). In this context, the proposed SWRL rules reason on the bearing individuals and infer about the bearing states of them. The SWRL rule in Fig. 8.7 is used for the identification of a Dir state:

The inference process of this SWRL rule is shown in Fig. 8.8. On the right side of the figure, the data property in the red rectangle is inferred by the launching of

*Bearing(?b) ^ hasBearingState(?b, ?s) ^ hasBearingParameter(?s, ?p1) ^ hasBearingParameter(?s, ?p2) ^ hasBearingParameter(?s, ?p3) ^ Kurtosis(?p1) ^ CrestFactor(?p2) ^ RMS(?p3) ^ hasParameterValue(?p1, ?v1) ^ hasParameterValue(?p2, ?v2) ^ hasParameterValue(?p3, ?v3) ^ swrlb:greaterThan (?v1, 3.266) ^ swrlb:greaterThan (?v2, 12.446) ^ swrlb:greaterThan(?v3, 20) -> **hasStateSpecification(?s, "Dir")***

Figure 8.7 – A SWRL rule for the identification of an inner race defect in a bearing.

this rule. This rule can be translated as: **If** a monitored bearing has a bearing state, and this bearing state is characterized by three parameters which are Kurtosis, Crest Factor, and RMS, and the values of these parameters satisfy the value ranges Kurtosis > 3.266, Crest Factor > 12.446, and RMS > 20 respectively, **then** the state of this bearing is identified as Dir.

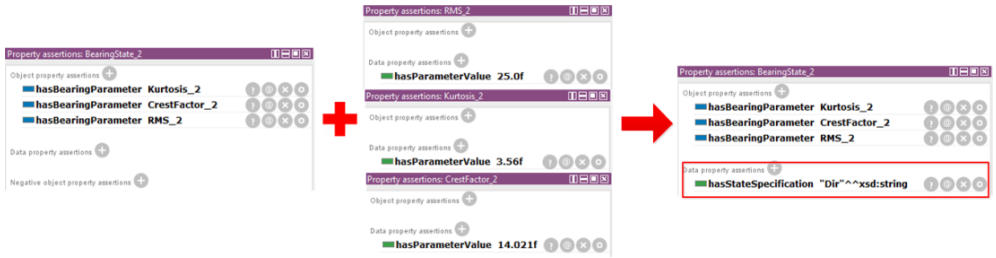



Figure 8.8 – The SWRL rule inference process for the identification of an inner race defect in a bearing.

Another example of a bearing defects detection task includes the consideration of temporal information. Normally, the behavior of a piece of machinery is represented as a combination of a sequence of states and the state transitions. During a machinery life cycle, the machinery condition degrades due to aging and wear, which decreases the performance of the machinery. This situation appeals to the need for analyzing machinery degradation trends, which allows maintenance activities to be scheduled. In this context, the SWRL rule in Fig. 8.9 is proposed to infer about the severity level of a bearing degradation trend.

*Bearing(?b) ^ hasBearingState(?b, ?s1) ^ hasParameter(?s1, ?p1) ^ Kurtosis(?p1) ^ hasParameterValue(?p1, ?v1) ^ swrlb:greaterThanOrEqual(?v1, 2.544) ^swrlb:lessThan(?v1, 5.158) ^ hasParameter(?s1, ?p2) ^ RMS(?p2) ^ hasParameterValue(?p2, ?v2) ^ swrlb:lessThan(?v2, 6) ^ temporal:hasTime(?s1, ?t1) ^ hasBearingState(?b, ?s2) ^ temporal:hasTime(?s2, ?t2) ^ swrlb:add(?t2, ?t1, 0.23) ^ hasParameter(?s2, ?p3) ^ Kurtosis(?p3) ^ hasParameterValue(?p3, ?v3) ^ swrlb:greaterThanOrEqual(?v3, 3.435) ^ swrlb:lessThan(?v3, 6.974) ^ hasParameterValue(?p4, ?v4) ^ RMS(?p4) ^ hasParameterValue(?p4, ?v4) ^ swrlb:greaterThanOrEqual(?v4, 6) -> **hasMinorError(?b, true)***

Figure 8.9 – A SWRL rule for the identification of a minor error in a bearing.

Fig. 8.10 shows the result of the SWRL rule inference process. The inferred data property in the red rectangle represents that the monitored bearing is identified as

having a minor-severity-level error. This rule can be interpreted as: if there is a bearing having a state at time t, and the state is characterized by the parameter Kurtosis and the parameter RMS, and the values of the two parameters satisfy the condition $2.544 \leq \text{Kurtosis} < 5.158$ and $\text{RMS} < 6$ respectively, and after 0.23 time units the same bearing experiences another state, which is also characterized by the parameter Kurtosis and the parameter RMS, and the values of the two parameters satisfy the condition $3.435 \leq \text{Kurtosis} < 6.974$ and $\text{RMS} \leq 6$, then this bearing is identified with a minor error.

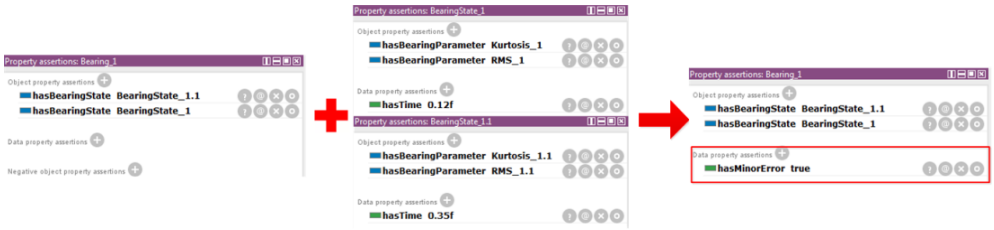



Figure 8.10 – The SWRL rule inference process for the identification of a minor error in a bearing.

The results of SWRL rule inferences show how the ontology could be used to reason about the machinery operation conditions. The reasoning capabilities of SWRL rules allow the condition monitoring tasks such as machinery state identification and error detection to be accomplished.

## 8.7 Ontology evaluation

Ontology evaluation enables users to assess the quality of ontologies. It is essential for the wide adoption of ontologies, since ontologies can be shared and reused by different users, and the quality of ontologies such as the consistency, completeness, and conciseness of taxonomies are key considerations when different users reuse ontologies in specific contexts. In this thesis, to evaluate the quality of the proposed MCMO ontology, we use OOPS!, which is an online ontology evaluation tool [PVGPSF14]. The reason we choose this tool for ontology evaluation is two-fold. Firstly, OOPS! allows automatic detection of common pitfalls in ontologies, and the detection of pitfalls can be executed independently of the ontology development software and platforms. Secondly, it enlarges the list of errors that can be detected by most recent ontology evaluation tools, thus providing a broad scope of anomaly detection in ontologies [PVGPSF14].

In OOPS!, ontology pitfalls are classified into three categories: structural, functional, and usability-profiling. Under each category, fine-grained classification criteria is provided to cope with specific types of anomalies.

MFPO and MCMO ontologies share common classes and relationships. Since MFPO is an extended version of the MCMO ontology with additional classes and relationships, the evaluation of MFPO ensures the accessement of MCMO at the same time. In this section, the MFPO ontology is examined according to the following three categories [PVGPSF14]:

- Structural dimension: it focuses on anomaly detection on syntax and formal semantics. Since the MCMO ontology consists of logical axioms, the syntax, and logical consistency can be evaluated and validated through anomaly detection within this category. To be more specific, This category is composed of five criteria: i). modeling decisions, which evaluates whether users use the ontology implementation language in the correct way; ii). real-world modeling or common sense, which evaluates the completeness of the domain knowledge formalized by the MCMO ontology; iii). no inference, which checks whether the desired knowledge can be inferred through ontology reasoning; iv). wrong inference, which refers to the detection of inference that leads to erroneous or invalid knowledge; and v). ontology language, which assesses the correctness of the ontology development language of the MCMO ontology.

- Functional dimension: it considers the intended use and functionality of the MCMO ontology. Under this category, two specific criteria are used to evaluate the MCMO ontology: i). requirement completeness, which evaluates coverage of the domain knowledge that is formalized by the MCMO ontology; ii). application context, which evaluates the adequacy of the MCMO ontology for a given use case or application.

- Usability-profiling dimension: it evaluates the level of ease of communication when different groups of users use the same ontology. Within this category, two specific criteria are applied for ontology evaluation: i). ontology understanding, which evaluates the quality of information or knowledge that is provided to users for easing the understanding of the ontology; ii). ontology clarity, which assesses the quality of ontology elements for being easily recognized and understood by users. These criteria are commonly used to check

the quality of ontologies when users do not have sufficient domain knowledge.

To evaluate the MCMO ontology according to the aforementioned categories, we uploaded the ontology code to the OOPS! online tool. After loading the ontology code, the ontology pitfall scanner is used to check the pitfalls that exist in the MCMO ontology. Fig. 8.11 shows the evaluation result. The result shows that our ontology is free of bad practices in the structural, functional, and usability-profiling dimensions of evaluation. Moreover, the MCMO ontology is developed and formalized using OWL, which is a widely used language for knowledge representation and ontology development. This eases the reuse of the MCMO ontology in other contexts and also simplifies the integration of the MCMO ontology with other knowledge components that are developed with the same language.

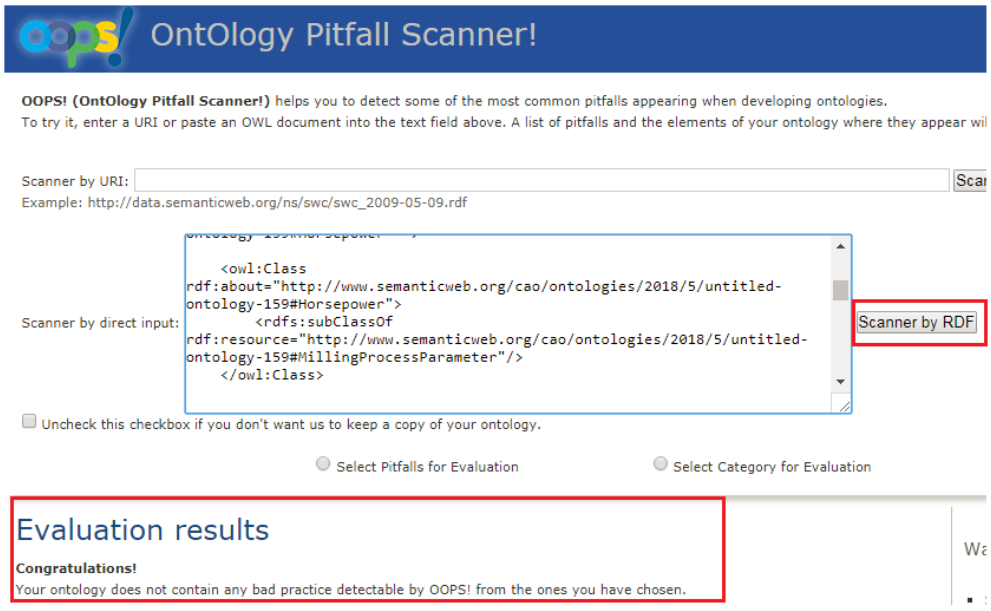



Figure 8.11 – Screen shot of the MCMO ontology validation results through OOPS!.

## 8.8 Summary

This chapter presents an ontological framework which is the basis for the development of a knowledge-based predictive maintenance system. The ontological framework is developed using a Middle-Out approach. It includes an ontology that models the core concepts and properties for condition monitoring, named CM-core.

The CM-core ontology contains core concepts related to condition monitoring, such as system, structure, behavior, function, state, fault and failure. The generality of the CM-core ontology ensures the possibility of its specialization into more specific domain ontologies, such as the ontologies for machine condition monitoring in industry. Thus, the CM-core ontology can be extended and customized to satisfy better the needs in individual domains where ontologies need to be used to empower information systems. The CM-core ontology is aligned with the foundational ontology UFO, for obtaining a rigorous conceptualization. Also, the CM-core ontology is specialized into domain ontologies, for integrating and representing domain-specific knowledge.

At the bottom layer of the framework, a domain ontology is presented for the formalization of knowledge that is related to condition monitoring tasks performed upon manufacturing processes. The ontology consists of three modules, named the *Manufacturing Module*, the *Context Module*, and the *Condition Monitoring Module*, respectively. After the construction of the ontology, we instantiate it with bearing examples in rotating machinery. A set of SWRL rules are proposed to reason about the operating states and error severity of the bearings, which enable the launching of condition monitoring activities.

# Chapter 9

# A novel hybrid semantic approach for predictive maintenance

## Contents

## 9.1 Introduction

In the context of predictive maintenance in smart factories, pattern mining has been widely used to discover frequently occurring temporally-constrained patterns, through which warning signals can be sent to humans for a timely intervention. Among pattern mining techniques, chronicle mining has been applied to industrial data sets for extracting temporal information of events and to predict potential machinery failures. However, even though chronicle mining results are expressive and interpretable representations of complex temporal information, domain knowledge is required for users to have a comprehensive understanding of the mined chronicles [PHW02]. As the predictive maintenance domain is becoming more knowledge-intensive, tasks performed in this domain can often benefit from incorporating domain and contextual knowledge, by which the semantics of the chronicle mining results can be explicitly represented and clearly interpreted. This helps to reduce the semantic gap issue, which stands for the gulf between the rich meaning and interpretation that users expect systems to associated with their queries to data, and the low-level features (e.g., attribute values) that systems actually compute. However, to the best of our knowledge, no work has been proposed to combine chronicle mining and semantics to facilitate the predictive maintenance of manufacturing processes. Also, most of the existing research works about predictive maintenance in the manufacturing domain merely focus on the classification of operating conditions of machines (e.g., normal operating condition, breakdown condition...), while lacking the extraction of specific temporal information of failure occurrence [SMS+19]. This brings obstacles for users to perform maintenance actions with the consideration of temporal constraints.

To address the aforementioned issues, in this chapter, a novel hybrid semantic approach is proposed. The semantic approach uses ontologies (introduced in Chapter 8) with their rule-based extensions to represent chronicle mining results in a semantic rich format, which enhances the representation and reuse of knowledge. By specifying domain semantics and annotating industrial data with rich and formal semantics, ontologies with their rule-based extensions help to address the issues described above. In more detail, the contributions of this chapter are as follows:

- This chapter proposes a novel hybrid semantic approach to automate machinery failure prediction tasks, which is based on the combined use of chron-

icles and semantic technologies.

- This chapter presents a novel algorithm to transform chronicles into SWRL-based logic rules, by which the predictive results are formalized, thus interpretable for both human and machines. The proposed transformation enables the automatic generation of SWRL rules from chronicle mining results, thus enabling an automatic semantic approach for machinery failure prediction.

- The feasibility and effectiveness of the proposed semantic approach is evaluated by conducting experimentation on a real industrial data set. The performance of SWRL rule construction and the quality of failure prediction is evaluated against the industrial data set.

## 9.2 Foundations and basic notions

This section introduces the foundations and basic notions of Sequential Pattern Mining (SPM) and chronicles that are necessary for describing our approach. The foundations include a formal description of the key concepts in these two research fields.

### 9.2.1 Foundations of sequential pattern mining

In industry, data collected for predictive maintenance tasks are normally represented as sets of timestamped sequences [SMS+19]. To cope with this type of data sets, SPM is one important technique to extract frequently occurring patterns. SPM is first studied by [AS+94], to analyze customer purchase behavior sequences. One SPM task could be described as follows: Given a data set containing a number of sequences, the goal of SPM is to find sequential patterns whose frequency of occurrence (a.k.a support) exceed a threshold fixed by an expert.

This support threshold indicates the minimal number of occurrences of the sequential patterns, and the found patterns are called frequent sequential patterns. For the output of SPM algorithms, each frequent sequential pattern is a sequence which consists of a set of items in a certain order.

To give a formal description of sequential patterns, this subsection reviews the definitions of key concepts. A sequence S is a set of ordered itemsets, denoted by

Table 9.1 – An example sequence data set.

| SID | Sequences |
|---|---|
| 10 | $< c(\underline{\mathbf{ab}}e)(a\underline{\mathbf{c}}f) >$ |
| 20 | $< (bcd)(ac)(bd)(adf)f >$ |
| 30 | $< (cd)(\underline{\mathbf{ab}})(b\underline{\mathbf{c}}f)e >$ |
| 40 | $< b(df)(bdf)c(ab) >$ |
| 50 | $< (ab)(bef)de >$ |
| 60 | $< (\underline{\mathbf{ab}}e)(\underline{\mathbf{c}}d)(ce) >$ |

$S =< SID, < I_1 I_2 I_3 ... I_n >>$, with SID standing for the index of the sequence with $I_j$ representing a non-empty set of items. Given two sequences $S_a =< SID_a, < a_1 a_2 a_3 ... a_m >>$ and $S_b =< SID_b, < b_1 b_2 b_3 ... b_n >>$, the sequence $S_a$ is considered to be the subsequence of $S_b$, denoted as $S_a \subseteq S_b$, if there exists integers $1 \leq k_1 < k_2 < ... < k_m \leq n$ such that $a_1 \subseteq b_{k1}$, $a_2 \subseteq b_{k2}$, ..., $a_m \subseteq b_{km}$ [SL13]. One example of sequence data set is shown in Table 9.1. In the table, each row is a sequence of elements. The elements are presented with a certain order, showing the precedence relationships among them. For example, regarding the definitions recalled before, the sequence $< ce(ac) >$ is the subsequence of $< \underline{\mathbf{c}}(ab\underline{\mathbf{e}})(\underline{\mathbf{ac}}f) >$. If the support is set to 3, we may check that $< (ab)c >$ is a sequential pattern with the support of 3.

Over the last decades, considerable contributions have been settled in the research field of SPM [FVLK+17]. As a result, various SPM algorithms have been proposed. Based on these proposed SPM algorithms, a variety of approaches and experiments have been launched to improve the performance and efficiency of SPM tasks.

### 9.2.2 Sequential pattern mining with time intervals

Even though sequential patterns contain information about the orders of items, the algorithms introduced in the previous subsection can not specify the time intervals between items. In real-world situations, the occurrences of events are often recorded with temporal information, such as time points and time intervals between events. Thus, several contributions have been proposed to obtain the time intervals between successive items in sequences. The notion of the time-interval sequential pattern is first presented by Yoshida et al. [YISI00]. The authors name this kind of patterns as *"delta patterns"*. A delta pattern is an ordered list of itemsets

with the time intervals between two neighboring itemsets. It can be represented as $A \xrightarrow{[0,3]} B \xrightarrow{[2,5]} C$, where $A \rightarrow B \rightarrow C$ is a frequent sequential pattern. The time intervals [0,3] and [2,5] are bounding intervals, which means the transition time of $A \rightarrow B$ is contained in the time interval [0,3], and the transition time of $B \rightarrow C$ is placed in the time interval [2,5].

With the introduction of delta patterns, a group of algorithms were proposed to facilitate the mining process in temporal sequence data sets. One significant contribution is the work by Hirate et al. [HY06]. In this work, the authors propose the Hirate-Yamana algorithm to mine all frequent time-extended sequences. To do this, the authors generalize SPM with item intervals. In the generalization, they define a set of time-extended sequences , denoted as $S_t =<$ SID, $(t_{1,1}, i_1), (t_{1,2}, i_2), (t_{1,3}, i_3), ..., (t_{1,n}, i_n) >>$, where $i_j$ means an item, and $t_{\alpha,\beta}$ is the item interval between items $i_\alpha$ and $i_\beta$, $t_{\alpha,\beta}$ can be interpreted according to two aspects of conditions [HY06]:

- If the data sets contain timestamps, which indicate the time of occurrences of items, then $t_{\alpha,\beta}$ becomes the time interval and can be computed by the equation $t_{\alpha,\beta} = i_\beta.time - i_\alpha.time$, where $i_\beta.time$ and $i_\alpha.time$ are timestamps of items $i_\alpha$ and $i_\beta$ respectively. For example, one time-extended sequence could be $< (0,c), (1,abe), (3,ac), (5,f) >$, which means item $c$ occurs at time point 0, followed by itemset $abe$ occurring at 1 time unit later. Itemset $ac$ occurs 2 time unites after $abe$, and the last itemset $f$ occurs 2 time unites after $ac$.

- If the data sets do not contain timestamps, then $t_{\alpha,\beta}$ may become the item gap and defined by the equation $t_{\alpha,\beta} = \beta - \alpha$. In this case, the item gap is defined as the number of items that occur between two items. This type of representation is suitable to be applied to data sets which contain fixed item intervals, but it is not applicable to data sets which contain various length of time intervals.

The study on existing notions and algorithms help to capture the core concepts in the domain of time-interval SPM. These core concepts form the foundations of chronicle mining.

### 9.2.3 Foundations of chronicles

As introduced in the previous subsection, the sequential patterns considered in this thesis are chronicles. Compared to normal sequential patterns that merely contain a set of ordered events, chronicles also describe the temporal constrains of these events. To give formal definition of chronicles, this section starts by introducing the concept of *Event*, given by [CMM12].

**Definition 2** (Event)**.** *Let $\mathbb{E}$ be a set of event types, and $\mathbb{T}$ a time domain such that $\mathbb{T} \subseteq \mathbb{R}$, where $\mathbb{R}$ is the set of real numbers. $\mathbb{E}$ is assumed totally ordered and is denoted $\leq_{\mathbb{E}}$. According to [CMM12], an event is a couple $(e,t)$ where $e \in \mathbb{E}$ is the type of the event and $t \in \mathbb{T}$ is its time. In SPM, events represent itemsets of a single sequence.*

A sequence contains a set of ordered events, which are timestamped. The events contained in a sequence appear according to their time of occurrences.

**Definition 3** (Sequence)**.** *Let $\mathbb{E}$ be a set of event types, and $\mathbb{T}$ a time domain such that $\mathbb{T} \subseteq \mathbb{R}$. $\mathbb{E}$ is assumed totally ordered and is denoted $\leq_{\mathbb{E}}$. According to the definition in [CMM12], a sequence is a couple $\langle \text{SID}, \langle (e_1,t_1),(e_2,t_2),...,(e_n,t_n)\rangle\rangle$ such that $\langle (e_1,t_1),(e_2,t_2),...,(e_n,t_n)\rangle$ is a sequence of events. $\forall i,j \in [1,n], i<j \Rightarrow t_i \leq t_j$. If $t_i = t_j$ then $e_i <_{\mathbb{E}} e_j$.*

When the events are time-stamped, how to describe the quantitative time intervals among different events is vital important for the prediction of possible future events. To achieve this goal, the notion *temporal constraints* is introduced in the following definition. The definition of *temporal constraints* is adopted from the one introduced in [CMM12].

**Definition 4** (Temporal constraint)**.** *A temporal constraint is a quadruplet $(e_1,e_2,t^-,t^+)$, denoted $e_1[t^-,t^+]e_2$, where $e_1,e_2 \in \mathbb{E}$, $e_1 \leq_{\mathbb{E}} e_2$ and $t^-,t^+ \in \mathbb{T}$.*

$t^-$ and $t^+$ are two integers which are called lower bound and upper bound of the time interval, such that $t^- \leq t^+$. A couple of events $(e_1,t_1)$ and $(e_2,t_2)$ are said to satisfy the temporal constraint $e_1[t^-,t^+]e_2$ iff $t_2 - t_1 \in [t^-,t^+]$.

It is defined that $e_1[a,b]e_2 \subseteq e_1'[a',b']e_2'$ iff $[a,b] \subseteq [a',b']$, $e_1 = e_1'$, and $e_2 = e_2'$.

With introducing the *events* and *temporal constraints* among different events within a sequence, the concept of chronicles [CMM12] is defined as follows.

**Definition 5** (Chronicle)**.** *A chronicle is a pair $\mathscr{C} = (\mathscr{E},\mathscr{T})$ such that:*

*1)* $\mathscr{E} = \{e_1 ... e_n\}$, *where* $\forall i, e_i \in \mathscr{E}$ *and* $e_i \leq_{\mathbb{E}} e_{i+1}$,

*2)* $\mathscr{T} = \{t_{ij}\}_{1 \leq i < j \leq |\mathscr{E}|}$ *is a set of temporal constraints on* $\mathscr{E}$ *such that for all pairs (i, j) satisfying* $i < j$, $t_{ij}$ *is denoted by* $e_i[t_{ij}^-, t_{ij}^+]e_j$.

$\mathscr{E}$ is called the episode of $\mathscr{C}$, according to the definition of episode's discovery in sequences [CMM12].

In the chronicle discovery process, *support* is used as a measure to compute the frequency of a pattern inside a sequence. It can therefore be formalized by the definition below.

**Definition 6** (Chronicle support)**.** *An occurrence of a chronicle* $\mathscr{C}$ *in a sequence S is a set* $(e_1, t_1)...(e_n, t_n)$ *of events of the sequence S that satisfies all temporal constraints defined in* $\mathscr{C}$ *[SMS+ 19]. The support of a chronicle* $\mathscr{C}$ *in a sequential data set* SD *is the number of its occurrences in* SD *with maximum one occurrence in each data sequence* S*, or the percentage of its occurrences (with maximum one occurrence in each* S*) over all data sequences in* SD.

The relevance of a chronicle is essentially based on the value of its support.

To illustrate these basic definitions, an example including a sequence and a chronicle extracted from it is presented. Assuming a sequence S contains three events < A, B, C >, represented as follows:

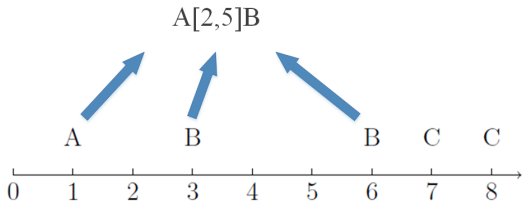



Figure 9.1 – A sequence representing three events.

In Fig. 9.1, time constraints that describe the pattern {A, B, C} are noted by A[2,5]B, B[1,5]C and A[6,7]C. Here [2,5], [1,4] and [6,7] lower and upper bounds of the time intervals among events.

After the generation of temporal constraints, these events can be represented as a graphical way, as shown in Fig. 9.2. In the figure, events are represented by the circles, and temporal constraints are displayed through arrows among events. The

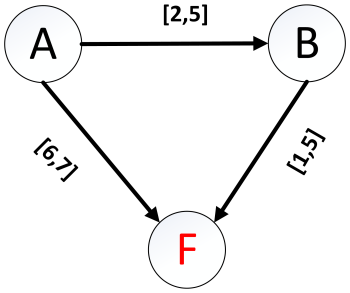



Figure 9.2 – Example of a chronicle.

values above each arrow are quantitative numerical bounds of temporal constraints.

In the domain of predictive maintenance, frequent chronicle mining has been used to detect machine anomalies in advance [SST18, SMS+19]. To combine frequent chronicle mining and semantics for facilitating predictive maintenance tasks, a special type of chronicles, called *failure chronicles* is introduced [SMS+19].

**Definition 7** (Failure chronicle)**.** *For a chronicle* $\mathscr{C}_{\mathrm{F}} = (\mathscr{E}, \mathscr{T})$, $\mathscr{C}_{\mathrm{F}}$ *is a failure chronicle if and only if the events that describe it are set according to their order of occurrence in the sequence, and within the chronicle there is a event that represents the failure, i.e. for* $\mathscr{E} = \{e_1 \cdots e_n | e_i \leq_{\mathbb{E}} e_{i+1}, i \in [1, n]\}$, $e_n$ *is the failure event.*

Based on Definition 7, we propose the formal definition of a *Chronicle rule*:

**Definition 8** (Chronicle rule)**.** *A chronicle rule is an implication* $\mathrm{R} : \mathrm{EC} \wedge \mathrm{TEC} \rightarrow \mathrm{TF}$ *such that:*

- $\mathrm{EC} = \{ec_1, ..., ec_n\}$, *which represents a set of non-failure events. The non-failure events in* EC *are ordered and denoted as* $\leq_{\mathrm{EC}}$. *For* $\forall i, ec_i \in \mathrm{EC}$, *and* $ec_i \leq_{\mathrm{EC}} ec_{i+1}$.
- $\mathrm{TEC} = \{t_{ij}\}_{1 \leq i < j \leq |\mathrm{EC}|}$ *is a set of temporal constraints on* EC *such that for all pairs* $(i, j$ *satisfying* $i < j$, $t_{ij}$ *is denoted by* $ec_i[t_{ij}^-, t_{ij}^+]ec_j$.
- $\mathrm{TF} = \{t_{ij}\}_{1 \leq i < k \leq |\mathrm{EC}+1|}$ *is a set of temporal constraints among the non-failure events* EC *and a failure event* F, *such that for all pairs* $(i, k)$ *satisfying* $i < k$, $t_{ik}$ *is denoted by* $ec_i[t_{ik}^-, t_{ik}^+]\mathrm{F}$.

In [SMS+19], a new algorithm called CPM has been introduced to mine frequent failure chronicles. Based on their work, in this chapter, a novel algorithm to automatically generate chronicle (SWRL) rules from frequent failure chronicles is pro-

posed. The generated SWRL rules aim to provide decision making for predictive maintenance in industry. The algorithm is introduced in Chapter 11.3.

## 9.3 A novel hybrid semantic approach for predictive maintenance

To propose the novel hybrid semantic approach for predictive maintenance, data mining and semantic technologies are jointly used, within which chronicle mining is used to predict the future failures of the monitored industrial machinery, and domain ontologies with their rule-based extensions are used to predict temporal constraints of failures and to represent the predictive results formally. The procedure of the semantic approach is shown in Fig. 9.3. Firstly, data pre-processing is implemented on raw industry data sets to obtain sequences in the form of pairs (event, timestamp), where each sequence finishes with the failure event. Secondly, a frequent chronicle mining algorithm mines the pre-processed data to discover frequent patterns that indicate machinery failures. Thirdly, based on the mined frequent patterns, semantic technologies are used to automate the generation of SWRL-based predictive rules. These rules enable ontological reasoning over individuals in ontologies, thus facilitating decision making.

### 9.3.1 Domain knowledge

Within an intelligent system, ontologies contain the domain knowledge to operate. In this work, the MFPO ontology introduced in Chapter 8.5 is used to describe the concepts and relationships within chronicles. The definitions of key concepts and

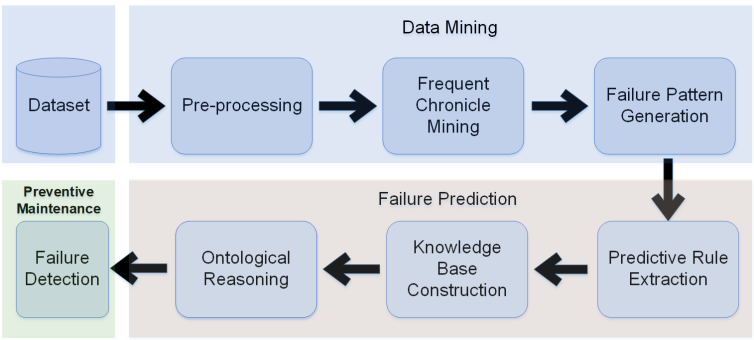



Figure 9.3 – The procedure of the semantic approach for predictive maintenance.

relationships in the MFPO ontology are formalized based on the basic notions introduced in Chapter 8 and Chapter 9.2.

### 9.3.2 Rules

In the proposed semantic approach, different SWRL rules are used for predicting machinery failures. The launching of these rules allows reasoning over individuals contained in the MFPO ontology. In this subsection, SWRL rules which are used to predict the time interval between a certain event and a future failure are introduced. After that, an algorithm developed for transforming chronicles into SWRL rules is demonstrated. The proposed rules and algorithm enable the automatic failure prediction in predictive maintenance tasks.

In this work, the reason SWRL rules are chosen is two-fold. Firstly, SWRL provides model-theoretic semantics and has the advantage of its close association with OWL ontologies, which enables the definition of complex rules for reasoning about individuals in ontologies. Secondly, the use of SWRL to write rules is independent of rule implementation languages within rule engines, which has the advantage of the flexible selection of rule engines and reasoning platforms. The generation of SWRL rules is based on the SPM mining results. As the mining of industrial data sets can generate frequent patterns which contain failure events, SWRL rules can be proposed to reason about the criticality of machinery failures. Therefore, when a failure is detected on a production line, SWRL rules can be launched to identify the criticality of this failure, which enables appropriate maintenance actions to be performed.

#### Failure time prediction rules

Even though the chronicle in Fig. 9.2 is represented in a structured format, it lacks formal semantics and domain knowledge to be interpreted by humans and predictive maintenance systems. For example, the meaning of events A, B, and C are missing, which may cause the semantic gap between chronicle mining results and users. To overcome this issue, ontologies with their rule-based extensions are used to represent chronicles in a semantic rich format, which helps the sharing and reuse of chronicle mining results.

As the mining of sequential data sets can generate frequent failure chronicles, SWRL rules can be proposed to reason about temporal information of machinery

failures. Therefore, when a new sequence of timestamped events arrives, SWRL rules can be launched to predict the time intervals among different events and future failures. As stated in Section 9.3.1, an event within a chronicle is determined by a set of observed properties (with their associated values). Based on this definition, the antecedent of such a rule is constructed by describing quantitative values of observed properties (attributes) and the temporal constraints inside a chronicle. The consequent of such a rule comprises the lower and upper bounds of the time intervals among certain events and the failure.

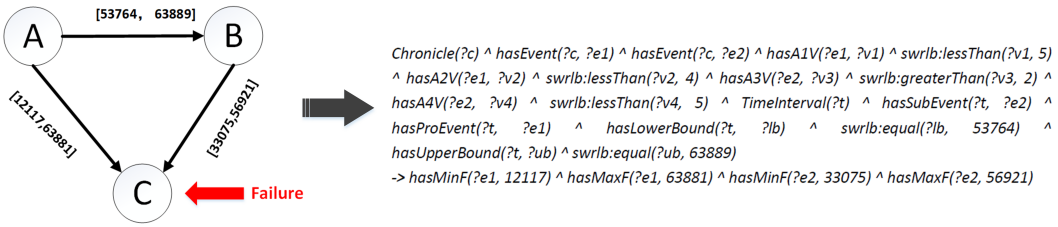



Figure 9.4 – Example of a SWRL-based predictive rule, generated from a chronicle.

Based on a failure chronicle, a SWRL rule can be elicited. Fig. 9.4 demonstrates how the rule that describes different events and temporal constraints can be constructed from a failure chronicle. Within the rule, *Chronicle* stands for the root class of all the chronicle individuals in the ontology. *hasEvent* is the object property that links individuals of the class *Chronicle* and those under the class *Event*. *hasA1V*, *hasA2V*, *hasA3V*, and *hasA4V* are data properties that assign quantitative values of attributes to the two individuals A and B under the *Event* class. *TimeInterval* corresponds to the root class of all individuals of time intervals. There are two object properties that link *TimeInterval* with *Event*: *hasSubEvent* and *hasProEvent*, among which *hasSubEvent* corresponds to the subsequent event of a time interval, and *hasProEvent* indicates the preceding event of a time interval. In this case, event A is the preceding event of the time interval between A and B, and event B is the subsequent event of this time interval. By describing the numerical values of different attributes and the time interval with its preceding and subsequent events, temporal constraints among events A, B with the failure C are indicated. The temporal constraints comprise the minimum time duration between an event with the failure, described by the data property *hasMinF*, and the maximum time duration between an event with the failure, described by another data property *hasMaxF*.

**Automatic rule generation based on chronicles**

To enable the automatic generation of a SWRL rule, in this work, a novel algorithm to transform chronicles into predictive SWRL rules is proposed. Algorithm 1 demonstrates the general idea of our rule transformation method. The $\wedge$ symbol stands for the conjunction operator for rule atoms. It runs in four major steps:

1) The function *LastNonfailureEvent* extracts the last non-failure event within a chronicle. As a chronicle contains a set of ordered events, the last non-failure event stands for the event to which the subsequent event is a failure.

2) For each temporal constraint in a chronicle, the two functions *PrecedingEvent* and *SubsequentEvent* extract the preceding and subsequent events of the time interval that is defined in this temporal constraint. Then the two events and this time interval forms different atoms in the antecedent of the rule, and they are treated as conjunctions.

3) For each last non-failure event before the failure (there could be multiple last events before the failure), extract the temporal constraint between this event and the failure. The extracted temporal constraint is treated as a conjunction with the last event, to form the consequent of the rule.

4) At last, a rule is constructed as an implication between the antecedent and the consequent.

A sequence can be described by one or multiple chronicles. To improve the quality of failure prediction, only the most relevant chronicles for the rule transformation are kept. In this context, features of chronicles such as *Chronicle Support* are selected as reference measures, to select the most relevant chronicles.

**Algorithm 1** Algorithm to transform a failure chronicle into a predictive SWRL rule.

**Input:** $\mathscr{C}_{\mathrm{F}}$: A chonicle model within which the last event is a failure event, $\mathscr{E}$: the episode of $\mathscr{C}_{\mathrm{F}}$ which contains different types of events in a chronicle.

**Output:** R

1: EL $\leftarrow$ L$ast$N$onfailure$E$vent(\mathscr{C}_{\mathrm{F}}, \mathscr{E})$ ▷ Extract the last non-failure event before the failure within a chronicle.
2: R $\leftarrow \emptyset$, A $\leftarrow \emptyset$, C $\leftarrow \emptyset$, A$tom_a \leftarrow \emptyset$, A$tom_c \leftarrow \emptyset$.
3: **for each** $e_i[t_{ij}^-, t_{ij}^+]e_j \in \mathscr{T}$ **do**
4: $pe \leftarrow$ P$receding$E$vent(e_i[t_{ij}^-, t_{ij}^+]e_j)$ ▷ Extract the preceding event of this time interval.
5: $se \leftarrow$ S$ubsequent$E$vent(e_i[t_{ij}^-, t_{ij}^+]e_j)$ ▷ Extract the subsequent event of this time interval.
6: A$tom_a \leftarrow [t_{ij}^-, t_{ij}^+] \wedge pe \wedge se$
7: A $\leftarrow$ A$tom_a \wedge$ A
8: **end for each**
9: **for each** $el \in$ EL **do**
10: $ftc \leftarrow$ F$ailure$T$imeConstraint(el,$ TI) ▷ Extract the time constraint between the last event before the failure and the failure event.
11: A$tom_c \leftarrow el \wedge ftc$
12: C $\leftarrow$ A$tom_c \wedge$ C
13: **end for each**
14: R $\leftarrow$ (A $\rightarrow$ C) ▷ A rule R is generated as an implication between the antecedent and consequent.
15: **return** R

## 9.4 Experimentation

To evaluate the effectiveness of our approach, a software prototype is developed based on Java 10.0.2, Protégé 5.5.0 [NCF+03], OWL API [HB11] and SWRL API [OKTM05]. The reason Protégé and OWL API are chosen is their convenience of creating, parsing, manipulating, and serializing OWL Ontologies. SWRL API allows us to create and interact with SWRL rules and SQWRL queries. Also, the graphical tools embedded in SWRL API ease the visualization and interpretation of rule-based reasoning and querying results. Among these tools, OWL API is used to build and manipulate the MFPO ontology. Different types of chronicles are created as individuals within the MFPO ontology, and SWRL-based predictive rules are proposed using the transformation algorithm introduced in Chapter 9.3.2. To enable ontology reasoning, the SWRL API, which includes a SWRL Rule Engine API, is used to create the transformed rules and then execute them. Within this process, the Drools rule

engine [Pro11] is used for rule execution. At last, the inferred knowledge is returned to the OWL API, and stored in the new ontology. The running environment of the software prototype is Microsoft Windows 10.

### 9.4.1 The SECOM data set

Our approach is validated by conducting experimentation on the SECOM data set [MJ08], which contains measurements of features of semi-conductor production within a semi-conductor manufacturing process. In the SECOM data set, 1567 data records and 590 attributes are collected, with each recording being characterized by a timestamp referring to the time that the data is recorded. Each recording is also associated with a label, which is either 1 or -1. The label of every recording explains the correctness of the event, with -1 corresponding to a non-failure event, and 1 refers to a failure. Timestamps are associated with all the records indicating the moment of each specific test point. In total, 104 number of records represent the failures of production. The data is stored in a raw text file, within which each line represents an individual example of recording with its timestamp. The features are separated by spaces.

However, the data contained in SECOM data set do not have the same types of attributes and values, that some of the information contained in the data is irrelevant to the failure prediction task thus is considered as noise. Moreover, due to the inter-dependency among individual features and the complex behavior of combined features, it is difficult to extract frequent patterns and rules based on the analysis of all the 590 attributes. Thus, in this context, instead of going through the entire data set and use all 590 attributes for failure prediction, feature selection methods [GE03] are used to identify and select the most relevant attributes in predicting the failures. The selected attributes are subsequently used to extract the key factors and patterns that lead to machine failures. This reduces the data processing time and memory consumption.

### 9.4.2 The extraction of frequent failure chronicles

The main objective is to extract frequent failure chronicles and test the performance of Algorithm 1 on the SECOM data set. To obtain frequent failure chronicles, the frequent chronicle mining approach introduced in [SMS$^+$19] is used. In [SMS$^+$19], an data pre-processing method is introduced, including data discretization and se-

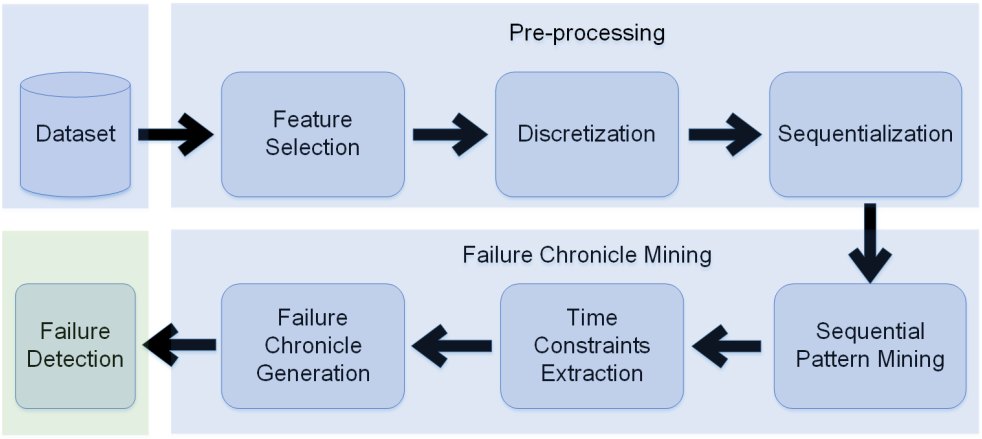



Figure 9.5 – Different steps used in the frequent failure chronicle mining approach, adapted from [SMS+19].

quentialization. Fig. 9.5 shows different steps within the data mining, especially the frequent chronicle mining approach. The steps presented in Fig. 9.5 elaborate the data mining procedure which is described in Fig. 9.3. The approach starts with the aforementioned feature selection method introduced in [GE03], after which a feature subset of the SECOM data set is obtained while retaining a suitably high accuracy in representing the original data set. As a result, 10 most relevant attributes are selected as the optimal subset of all 590 attributes. After the feature selection, data discretization [RGGMT+16] is employed to discretize continuous values for obtaining nominal ones. Thereafter, data sequentialization is used to transform the data into the form of pairs (event, timestamp), where each sequence finishes with a failure [SMS+19]. After that, the ClaSP algorithm [GCMG13] is applied to data sequences, to extract frequent sequential patterns. Also, the frequent chronicle mining algorithm introduced in [SMS+19] is used to extract the temporal constraints among these sequential patterns. Up to this step, frequent failure chronicles are obtained and transformed into predictive rules.

As introduced in Chapter 9.3, to improve the quality of failure prediction, *Chronicle Support* is chosen as a reference measure, to select the most relevant failure chronicles for failure prediction. As a result, only a subset of all frequent chronicles are used for predictive rule transformation. Table 9.3 shows the failure chronicles that have the 10 highest chronicle support. These chronicles are used as examples to demonstrate the predictive rule generation approach. In Table 9.3, each failure chronicle is described by the number of events that it contains, the number of time intervals among events, all the observed properties (attributes) that characterize the

failure chronicle, and the chronicle support. For the ease of demonstration, the 590 attributes are labeled as $A_1$, $A_2$, $A_3$...$A_{590}$.

For an event within a failure chronicle, it is not only identified by a set of attributes, but also the quantitative values of them. To obtain the corresponding quantitative attribute values for describing each event, data discretization has been applied to the SECOM data set. After data discretization, the quantitative data has been translated into qualitative data. Also, an association between each numerical value and a certain interval has been created. Taking the chronicle that is presented in Fig. 9.4 as an example, Table 9.2 shows the numerical intervals for describing the events within this failure chronicle. This chronicle is the failure chronicle $\mathscr{C}_{F5}$ introduced in Table 9.3.

Table 9.2 – Attributes with their numerical intervals within the failure chronicle $\mathscr{C}_{F5}$.

| Event | Attribute | Numerical Value Interval |
|---|---|---|
| A | 63 | [89.2564, 94.8757) |
| A | 204 | [4925.1678, 4999.2456) |
| A | 209 | [20.1884, 23.0750) |
| A | 347 | [6.4877, 6.9573) |
| A | 476 | [125.1988, 137.4435) |
| B | 58 | [4.5537, 4.8994) |
| B | 63 | [89.2564, 94.8757) |
| B | 64 | [90.0196, 94.3934) |
| B | 102 | [-0.1188, 0.5231) |
| B | 347 | [6.2446, 6.9574) |

### 9.4.3 The generation of SWRL-based predictive rules

Based on the descriptions of the failure chronicle $\mathscr{C}_{F5}$, the algorithm introduced in Section 9.3.2 is used to generate a SWRL-based predictive rule. The result of this rule generation is shown in Fig. 9.6. In this rule, *hasA58V*, *hasA63V*, *hasA64V*, *hasA102V*, *hasA204V*, *hasA209V*, *hasA347V*, *hasA476V* are data properties in the MFPO ontology that link individuals of the *Event* class with XML Schema Datatype values. They correspond to the quantitative values of the attributes $A_{58}$, $A_{63}$, $A_{64}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{347}$, and $A_{476}$ in the SECOM data set. To describe the numerical intervals which are obtained by discretization, SWRL Built-Ins are used to specify the upper and lower numerical boundaries. The consequent of this rule comprises the temporal constraints among *Events* A, B and C. The minimum time duration between an event

with the failure is described by the data property *hasMinF*, and the maximum time duration between an event with the failure is described by another data property *hasMaxF*. By this way, the temporal constraints of a future failure is inferred by the launching of such a predictive SWRL rule. This rule is an instantiation of the generic rule introduced in Fig. 9.4.

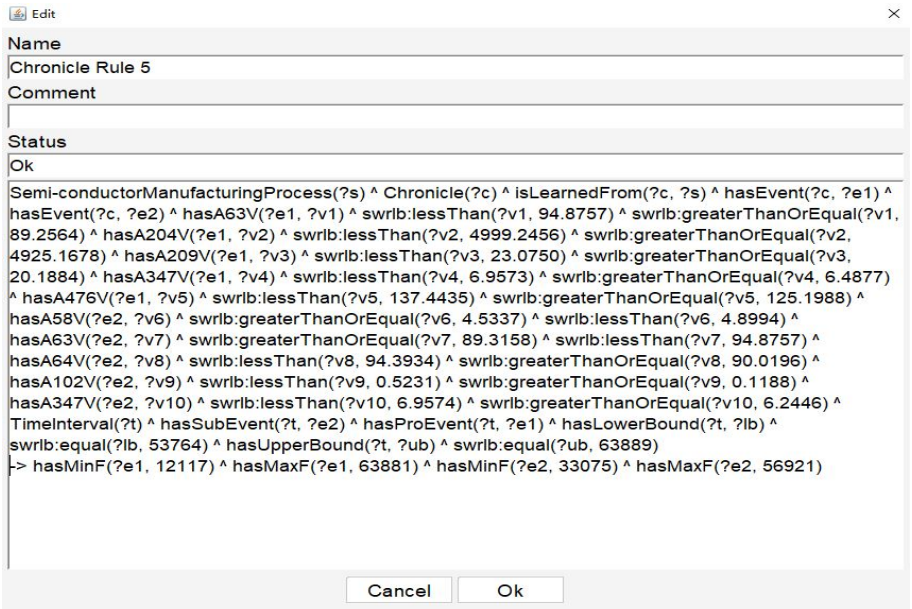



Figure 9.6 – The SWRL-based predictive rule transformed from the failure Chronicle $\mathscr{C}_{\mathrm{F5}}$, with describing the attributes and their numerical value intervals.

Table 9.3 – Extracted failure chronicles that have the highest 10 chronicle support.

| Failure Chronicle | Number of Events | Number of Time Intervals | Attributes | Chronicle Support |
|---|---|---|---|---|
| $\mathscr{C}_{F1}$ | 3 | 3 | $A_{63}$, $A_{64}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{476}$ | 83.65% |
| $\mathscr{C}_{F2}$ | 3 | 3 | $A_{63}$, $A_{64}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{347}$, $A_{476}$ | 82.69% |
| $\mathscr{C}_{F3}$ | 3 | 3 | $A_{58}$, $A_{64}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{476}$ | 82.69% |
| $\mathscr{C}_{F4}$ | 3 | 3 | $A_{58}$, $A_{63}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{347}$ | 81.73% |
| $\mathscr{C}_{F5}$ | 3 | 3 | $A_{58}$, $A_{63}$, $A_{64}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{347}$, $A_{476}$ | 81.73% |
| $\mathscr{C}_{F6}$ | 3 | 3 | $A_{58}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{347}$, $A_{476}$ | 80.77% |
| $\mathscr{C}_{F7}$ | 3 | 3 | $A_{58}$, $A_{204}$, $A_{209}$, $A_{347}$, $A_{476}$ | 80.77% |
| $\mathscr{C}_{F8}$ | 4 | 4 | $A_{63}$, $A_{64}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{347}$, $A_{476}$ | 78.84% |
| $\mathscr{C}_{F9}$ | 4 | 4 | $A_{58}$, $A_{63}$, $A_{102}$, $A_{204}$, $A_{209}$, $A_{347}$ | 78.84% |
| $\mathscr{C}_{F10}$ | 4 | 4 | $A_{58}$, $A_{204}$, $A_{209}$, $A_{347}$, $A_{476}$ | 78.84% |

## 9.5 Results evaluation

To evaluate the usefulness and effectiveness of our approach, an evaluation for SWRL rule-based failure prediction is conducted. It should be noted that the focus of results evaluation is on the quality of semantic enrichment to the chronicle mining results, and the evaluation of the performance of the chronicle mining phase is out of the scope of this thesis.

To perform results evaluation, the SWRL rules are applied to the data sequences in the SECOM data set, and three measures are used to assess the quality of these rules: the *True Positive Rate (TPR)*, the *Precision* of failure prediction, and the *F-measure*. The equations for computing these three measures are shown in Equation 9.1, 9.2 and 9.3. The number of generated SWRL rules are shown in Table 11.3.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{9.1}$$

$$\mathrm{P}recision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \tag{9.2}$$

$$\mathrm{F} - measure = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}. \tag{9.3}$$

Among them, the *True Positive Rate* aims to measure the percentage of positive sequences that have been correctly classified. In Equation 9.1, TP (True Positive) is the true positive results standing for the number of valid sequences that at least one SWRL rule could predict the failures in these sequences, and FN (False Negative) is the false negative results which stand for the number of sequences that no SWRL rule could predict the failures in these sequences.

The *Precision* of failure prediction measures the percentage of sequences based on which the SWRL rules are constructed correctly. For a given sequence, failure chronicles are extracted through chronicle mining and SWRL rules are constructed for failure prediction. After applying the SWRL rules, if the predicted failure temporal constraints are out of the range of the failure occurrence time intervals in the sequence, then it indicates that the SWRL rules could not predict the temporal constraints of the failure in this sequence. Thus, the failure is classified as *False Positive*. In Equation 9.2, TP is the same definition as in Equation 9.1, and FP (False Positive) is the number of sequences for which the SWRL rules incorrectly predict the temporal constraints of the future failures.

With obtaining the above two measures, the *F-measure* can be computed according to the Equation 9.3.

Table 9.4 shows the experimental results of the three measures. The three measures are computed according to different support thresholds of sequences in the data set. $ft_{min}$ is used to denote the support of a chronicle/SWRL rule in the data set.

Table 9.4 – *True Positive Rate, Precision* and *F-measure* of Failure Prediction Based on SWRL Rules.

| $ft_{min}$ | True Positive Rate | Precision | F-measure |
|---|---|---|---|
| 1 | 83.63% ±6.43% | 84.62% ±6.55% | 86.55% ±4.89% |
| 0.9 | 85.45% ±4.98% | 87.49% ±6.16% | 88.54% ±6.06% |
| 0.8 | 87.27% ±7.50% | 84.58% ±6.55% | 85.71% ±6.98% |
| 0.7 | 89.09% ±6.68% | 86.22% ±6.43% | 87.52% ±6.51% |
| 0.6 | 90.90% ±7.93% | 88.71% ±5.26% | 89.21% ±5.43% |
| 0.5 | 90.90% ±7.93% | 86.83% ±4.41% | 87.88% ±5.77% |

It can be seen from Table 9.4 that all computed values for the three measures are above 80%, which shows the results are encouraging. As the support $ft_{min}$ values decreases, the values of three measures show an increase tendency. This can be explained as follows: as $ft_{min}$ increases, the number of extracted chronicles decreases, which lead to the decrease of the number of transformed SWRL rules. For this reason, each sequence for testing is less likely to be validated by the transformed SWRL rules.

Since the SWRL rules are generated from chronicle mining results, the quality of their prediction exclusively depend on the mined frequent chronicles. In this context, the 10-fold cross validation principle [AC+10] is used to evaluate the quality of failure prediction. To apply the 10-fold cross validation principle, the SECOM data set is partitioned into two parts: the training set and the test set. Firstly, chronicles are extracted from the training sequences in the training set. Then, for the test set, it is checked for each sequence, its membership in at least one chronicle among those extracted. The number of sequences validated by the chronicles is computed to estimate its percentage of occurrence with respect to the whole data set. The computation of *True Positive Rate*, *Precision*, and *F-measure* follows the same principle for computing the recall rate, as introduced in [Sto74]. This procedure is repeated 10 times to validate all the sequences of the database.

The launching of such a set of SWRL-based predictive rules enables the predic-

tion of temporal constraints of future machinery failures. This allows users to take further maintenance actions, such as the replacement of the machine tools used on the production line. The performance of failure prediction could be enhanced by considering a new set of rules that reason about the criticality levels of failures. In the next chapter, machine learning techniques are used to classify the criticality levels of failures, according to the temporal constraints among the failures and other events.

## 9.6 Summary

This chapter demonstrates a novel hybrid semantic approach for implementing predictive maintenance in industry. The proposed semantic approach is a combination of frequent chronicle mining and semantics, within which chronicle mining is used to extract frequent chronicles based on industrial data sets, and a knowledge-based structure is used to automate the SWRL rule generation process and to formalize the predictive maintenance results.

The contributions of this chapter are three-fold. Firstly, chronicles are formally represented with the use of ontologies and SWRL rules, by which the main concepts and relationships for describing chronicles are formalized, then easing the knowledge representation and interpretation of frequent chronicle mining results. Secondly, a novel algorithm for transforming chronicles into SWRL-based predictive rules is introduced. The novel algorithm allows the automatic generation of SWRL rules based on the mined frequent chronicles, thus enabling an automatic semantic approach for predictive maintenance. Thirdly, the reasoning about temporal constraints of future machinery failures is enabled by the joint use of data mining and semantics, which allows the implementation of maintenance actions such as alarm launching.

However, the main problem of the proposed approach is the evolution of the ontology and the rule base. Since the manufacturing domain is highly-dynamic, the predictive maintenance system should be able to adapt itself to dynamic situations over time, for example, the change of context. Also, when the system fails to provide satisfactory results through launching the rules, it is required to consult domain experts for decisions about failure prediction and maintenance. In this situation, the domain experts use their expertise and experience to assess the current state of the system and provide appropriate decisions. For example, when the tem-

perature measured by a sensor located at a cutting tool exceeds its threshold and no rule in the rule base is able to warn about his abnormal condition, domain experts can use their experience to identify this abnormal condition and provide possible solutions in order to avoid the production line to produce unqualified products. In this way, new rules which capitalize experts' experience need to be proposed to update the initial set of rules in the rule base, in order to facilitate the quality of failure prediction. In this context, when the next time a similar situation needs to be addressed, the rule which capitalizes domain experts' experience will be launched together with the initial rules to identify potential failures and to make predictions. This requires the ontology and the rule base to be capable of coping with the dynamic change of knowledge. To cope with this issue, rule base update and refinement solutions should be proposed: the rule base should be updated and refined with each time of integration for one expert rule. During the integration process, the conflict issue among different chronicle rules should also be addressed. In this thesis, a rule base refinement approach is proposed to deal with these issues. This part of contributions will be introduced in Chapter 11.

# Chapter 10

# Assessment of machinery failures based on criticality levels

## Contents

## 10.1 Introduction

In the predictive maintenance domain, the prediction and assessment of failure criticality is a critical issue for manufactures. By obtaining criticality levels of different failures, machine operators can prioritize maintenance actions for higher criticality-level failures compared to lower level ones. In this way, the failure prediction results can be further used for scheduling and planning the maintenance of machines. However, existing predictive maintenance approaches in the manufacturing domain are limited to the deployment of condition monitoring systems for detecting anomalies and predicting the time of future machinery failures, while lacking the solutions for identifying the criticality of machinery failures [AGN19]. This causes a missing link between the temporal information of an anomaly (e.g., the occurrence time of a future machinery failure) and the criticality of the anomaly. To assess the availability of computing systems, the outage duration is a key consideration which indicates the criticality of a mechanical failure [ALRL04]. Therefore, predicting the moments of machinery failures are crucial for computing the outage duration and the criticality of the failures. In Chapter 9, a hybrid semantic approach for predicting the time of failures has been introduced. However, the proposed approach and existing research works fail to support decision makings about the criticality of machinery failures based on the time of occurrences of future machinery failures. This brings obstacles to users for performing appropriate maintenance actions with considering time limitations.

On the other hand, since most of the existing ontologies and rule-based approaches are based on crisp logic, they are not competent in dealing with uncertain situations in predictive maintenance. Comparing to approaches that use crisp logic, a fuzzy approach allows better expression of imprecise relations. For the scheduling of maintenance, using a fuzzy approach allows to associate each failure criticality assessment with a fuzzy index, indicating the degree of its membership to a “low" or “high" criticality level. By this, the crisp logic-based rules can be transformed into fuzzy rules, which enhances the representation of imprecise criticality level of machinery failures. In this way, the fuzzy approach provides a better solution for solving the symbol anchoring problem [CS03] than crisp logic-based approaches. Moreover, as the degradation process of a piece of machinery often involves inherent randomness, techniques that can handle uncertainty are required to avoid the outage situation of the machinery and to ensure the smooth operation of the pro-

duction system.

In this chapter, the work of Chapter 9 is extended by introducing two hybrid ontology-based approaches for the failure prediction and clustering tasks. The hybrid ontology-based approaches are based on the combined use of clustering techniques and ontology reasoning. Within the proposed approaches, we address the uncertainty of failure criticality by adopting two uncertainty frameworks: fuzzy *c*-means (FCM) clustering [BEF84] and evidential *c*-means (ECM) clustering [MD08b]. These tools are used to assess failures according to their criticality levels. On the other hand, domain ontologies and a set of SWRL predictive rules are proposed to reason about the time and criticality levels of machinery failures. The approach has been validated on one real-world industrial data set and several synthetic data sets.

## 10.2 The use of fuzzy clustering and ontology reasoning for machinery failure clustering

The first machinery failure clustering approach is based on the use of FCM clustering and ontology reasoning. It starts with the implementation of sequential pattern mining (SPM) [AS+95] on raw industrial data sets, after which frequent sequential patterns are obtained. The obtained frequent sequential patterns contain failure events as well as the temporal information of these failures (e.g., the time stamp indicating when the failure will happen). After that, FCM clustering is applied to the extracted temporal information of failures, in order to cluster different failures according to their time of occurrence. This clustering enables the identification of failure criticality. Then, domain ontologies with their rule-based extensions are used to formalize the domain knowledge and enable the prediction of the time and criticality of future failures. In this work, the failure chronicles that are in a rule format[1] are considered. This allows us to use these sequential patterns for ontology reasoning, to facilitate failure prediction in industry. Fig. 10.1 shows the steps described above.

### 10.2.1 Fuzzy clustering of failure criticality

In Chapter 8, one issue encountered is the incorrect identification of failure criticality. The incorrect identification is due to the reasoning with crisp logic, which fails

[1] In this work we consider rules in the format A $\xrightarrow[[t_1,t_2]]{}$ F$ailure$ where A is the antecedent part of the rule that predicts a failure in $[t_1, t_2]$ time units.

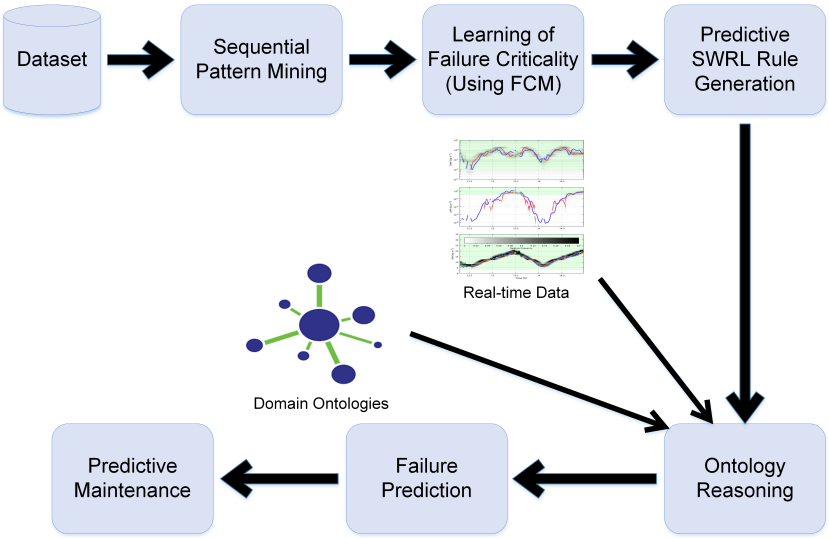



Figure 10.1 – Different steps of the hybrid ontology-based approach for predictive maintenance (using FCM and ontology reasoning).

to identify the criticality of a failure into the correct category when there are uncertain situations. For example, if a failure is predicted to happen within a considerably short amount of time after a "normal" condition, then the criticality level of this failure is identified as "high", meaning that maintenance actions need to be proposed immediately. However, if the predicted time duration between a "normal" condition and a failure falls into the "medium" category and the time duration is considerably close to the numerical threshold between "medium" and "high", then the system failed to identify the criticality of this failure into the correct category. To cope with this issue, a fuzzy approach which is able to handle such type of uncertainty situations is required. It should be noted that the failure criticality descriptor considered in this section is the time duration among normal conditions and machinery failures. Other descriptors that may affect the failure criticality, such as the level of mechanical component fracture and wear are out of the scope of this section.

To cope with the aforementioned uncertain situations, in this section, the FCM algorithm is used for fuzzy clustering of failure criticality. After executing the SPM step introduced in Chapter 9.3, the frequent sequential patterns which contain temporal information (e.g., the time duration among "normal" events and the failure events) of failure events are extracted. The temporal information is represented as data points during the implementation of the FCM algorithm. Then the FCM algorithm is used to cluster the failures by grouping similar data points into clusters. This clustering is achieved by iteratively minimizing an objective function which is

dependent on the distance of the data points to the cluster centers. The objective function is computed by the following equation:

$$\mathrm{J} = \sum_{j=1}^{\mathrm{N}} \sum_{i=1}^{\mathrm{C}} u_{ij}^{m} || \, x_j - c_i \, ||^2, \tag{10.1}$$

where $u_{ij}$ represents the degree of membership of the data point $x_j$ in the $i$th cluster, $c_j$ stands for the d-dimension center of the cluster, $m$ is any real number greater than 1, and $|| \cdot ||$ denotes any norm expressing the similarity between any measured data and the center.

During the implementation of FCM algorithm, the objective function is minimized with the update of membership $u_{ij}$ and the cluster centers $c_j$. The update of membership of the objective function is described by the following equation:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} (\frac{||x_i - c_j||}{||x_i - c_k||})^{\frac{2}{(m-1)}}}. \tag{10.2}$$

The update of the cluster centers $c_j$ is computed by:

$$c_j = \frac{\sum_{i=1}^{\mathrm{N}} u_{ij}^{m} \cdot x_i}{\sum_{i=1}^{\mathrm{N}} u_{ij}^{m}}. \tag{10.3}$$

By this, the FCM algorithm starts with an initial guess for cluster centers, and then iteratively updates the cluster centers and the degrees of membership for each data point. The iteration stops when $max_{ij} | \, u_{ij}^{(k+1)} - u_{ij}^{k} \, | < \varepsilon$, where $k$ is the number of iteration steps, and $\varepsilon$ is a termination criterion [BEF84]. The whole process converges when a local minimum value of J is obtained. The full theory of FCM clustering algorithm is presented in Appendix A.

In this work, the time duration from normal conditions to failures are used as training examples to FCM with three fixed clusters, and the three clusters indicate three levels of failure criticality:

- High Criticality: time from a normal condition to the failure is relatively short and the production line should be stopped for immediate maintenance.
- Medium Criticality: the failure may happen after a moderate amount of time and machine operators need to plan maintenance in a limited time.
- Low Criticality: the failure may happen in the long future and machine operators will have sufficient time to plan maintenance actions.

The results of the FCM algorithm are formalized by domain ontologies and SWRL rules, which aim to predict the time of future failures and their criticality, thus facilitating the knowledge representation and interpretation in predictive maintenance of industrial manufacturing process.

To handle uncertainty, the MFPO ontology (introduced in Chapter 8) has been extented. Normally, classical ontologies apply classical set theory, which indicates that elements either belong to a set or not [BS11]. This leads to the inability of classical ontologies for representing vague knowledge. This issue can be addressed by the fuzzy set theory, in which elements can belong to a set to some degree. To give a formal description, let us assume X a set of elements and $x$ is an element within the set X. In classical set theory, 0 means no-membership for a $x \in$ X and 1 means full membership. While in fuzzy set theory, a fuzzy subset of X, denotated as F, is defined by a membership function F(X). F(X) assigns any $x \in$ X to a value in the interval of real numbers between 0 and 1 [BS11]. By this, the criticality of a failure can be assess with multiple categories simultaneously, with each category assigned with a degree of membership.

In this extension of the MFPO ontology, the nominal categories of classes are described by data properties. In this way, the classes in the MFPO ontology are associated with membership degrees which range from 0 to 1. For example, in the MCMO ontology introduced in [CGZM+19], *hasFailureCriticality* is an object property whose domain is the class *Failure*, and range is the predefined individuals *Low*, *Medium* and *High*. After applying the aforementioned method, this object property is replaced by three data properties: *hasFailureCriticalityLow*, *hasFailureCriticalityMedium*, and *hasFailureCriticalityHigh*, and the sum of the numeric values of these three data properties is 1. By this, the *Failure* class in the MFPO ontology is associated with membership degrees to three nominal categories, thus enabling ontology reasoning of vague knowledge about this class.

### 10.2.2 Ontology reasoning for failure prediction

To predict the time of failures and also assess failure criticality with different categories based on their temporal information, SWRL rules are proposed for ontology reasoning. The proposed SWRL rules reason on the individuals in the MFPO ontology, and infer new knowledge about failure prediction.

As presented in Fig. 8.6, the *State* of a *ManufacturingResource* is determined by a

set of *ObservableProperties* (with their associated values). Based on this definition, the antecedent of a SWRL rule is constructed by describing quantitative values of *ObservableProperties* (attributes) and the temporal information of the failure. The consequent of such a rule comprises the time constraints of the failure and its criticality.

After obtaining the failure chronicles, to enable the automatic generation of SWRL rules, this subsection proposes a novel algorithm to transform failure chronicles (such as the one in Fig. 9.2) into predictive SWRL rules. The transformed SWRL rules not only predict the temporal constraints of failures, but also gives the memberships of failure criticality to differnet clusters (High, Medium, Low). The proposed algorithm is an extension of the algorithm 1 in Chapter 9. Algorithm 2 demonstrates the general idea of our rule transformation method. The $\wedge$ symbol stands for the conjunction of rule atoms. It runs in five major steps:

1) The function *LastNonfailureState* extracts the last non-failure state (event) within a chronicle, and the function *LastNonfailureState* extracts the failure within a chronicle.

2) For each temporal constraint in a chronicle, the two functions *PrecedingEvent* and *SubsequentEvent* extract the preceding and subsequent events of the time interval that is defined in this temporal constraint. Then the two events and this time interval forms different atoms in the antecedent of the rule, and they are treated as conjunctions.

3) For the last non-failure state before the failure, extract the time interval and its duration between this state and the failure. Then the descriptions of all normal states and this time duration form different atoms in the consequent of the rule, and they are treated as conjunctions. The extracted temporal interval and its duration is treated as a conjunction with the last event, to form the consequent of the rule.

4) FCM algorithm is applied to assess the failures according to their criticality. The failures are assigned to three categories, and three object properties in the MFPO ontology are used to represent the degrees of membership to different clusters. The degrees of membership are treated as a conjunction, to form the consequent of the rule.

5) At last, a rule is constructed as an implication between the antecedent and the consequent.

**Algorithm 2** Algorithm to transform a chronicle into a predictive SWRL rule, based on fuzzy-$c$ means

**Input:** $\mathscr{S}_{\mathrm{F}}$: a chronicle within which the last state (event) is a failure, $\mathscr{E}$: a set of the states that are described within a chronicle.

**Output:** R ▷ R: the SWRL rule to be constructed.

1: $ls \leftarrow \mathrm{L}ast\mathrm{N}onfailure\mathrm{S}tate(\mathscr{S}_{\mathrm{F}}, \mathscr{E})$ ▷ Extract the last non-failure state before the failure within a chronicle.

2: $f \leftarrow the\mathrm{F}ailure(\mathscr{E})$ ▷ Extract the failure within a chronicle.

3: $\mathrm{R} \leftarrow \emptyset$, $\mathrm{A} \leftarrow \emptyset$, $\mathrm{C} \leftarrow \emptyset$, $\mathrm{A}tom_a \leftarrow \emptyset$, $\mathrm{A}tom_c \leftarrow \emptyset$, $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{L}ow} = 0$, $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{M}edium} = 0$, $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{H}igh} = 0$. ▷ A: the antecedent of the SWRL rule. C: the consequent of the SWRL rule. $\mathrm{A}tom_a$: a subset of all atoms within the antecedent. $\mathrm{A}tom_c$: a subset of all atoms within the consequent. $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{L}ow}$, $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{M}edium}$, $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{H}igh}$: the three degrees of membership to the three categories of failure criticality.

4: **for each** $e_i \in \mathscr{E}$ **do**

5: $pe \leftarrow \mathrm{P}receding\mathrm{S}tate(e_i, \mathscr{S}_{\mathrm{F}})$ ▷ Extract the preceding state

6: $se \leftarrow \mathrm{S}ubsequent\mathrm{S}tate(e_i, \mathscr{S}_{\mathrm{F}})$ ▷ Extract the subsequent state

7: $\mathrm{A}tom_a \leftarrow pe \wedge se$

8: $\mathrm{A} \leftarrow \mathrm{A}tom_a \wedge \mathrm{A}$

9: **end for each**

10: $ftd \leftarrow \mathrm{F}ailure\mathrm{T}ime\mathrm{D}uration(ls, f)$ ▷ Extract the time duration between the last non-failure state and the failure.

11: $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{L}ow} \leftarrow \mathrm{D}egree of \mathrm{M}embership \mathrm{L}ow(ftd)$

12: $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{M}edium} \leftarrow \mathrm{D}egree of \mathrm{M}embership \mathrm{M}edium(ftd)$

13: $\mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{H}igh} \leftarrow \mathrm{D}egree of \mathrm{M}embership \mathrm{H}igh(ftd)$ ▷ Use the FCM algorithm to compute the degrees of membership of this time duration to the three clusters.

14: $\mathrm{C} \leftarrow \mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{L}ow} \wedge \mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{M}edium} \wedge \mathrm{F}_{\mathrm{F}ailure\mathrm{C}riticality\mathrm{H}igh} \wedge ftd$

15: $\mathrm{R} \leftarrow (\mathrm{A} \rightarrow \mathrm{C})$ ▷ A rule is generated as an implication between the antecedent and consequent.

16: **return** R

## 10.3 The use of evidential clustering and ontology reasoning for machinery failure clustering

Although FCM uses fuzzy partition methods that allow each object data to belong to two or more clusters with different degrees of membership, it is afflicted by the poor robustness against noise and outliers. To overcome this drawback of FCM, the evidential c-means (algorithm) was proposed. ECM extends the concept of fuzzy partition in FCM by allocating, for each object, a "mass of belief", not only to single clusters, but also to any subsets of $\Theta = \{\omega_1, \omega_2, ..., \omega_K\}$ [MD08b]. In this way, ECM has additional flexibility than FCM that allows users to gain a deeper insight from the data. Also, ECM improves the robustness against noise and outliers.

This section studies the performance of ECM when applied to the assessment of machinery failure criticality.

### 10.3.1 The evidence theory

The evidence theory [Dem67, SK94] is based on several fundamentals such as the *Basic Belief Assignment* (BBA). A BBA $m$ is the mapping from elements of the power set $2^\Theta$ onto [0, 1]:

$$m : 2^\Theta \longrightarrow [0,1],$$

where $\Theta$ is the *frame of discernment*. It is the set of possible answers for a treated problem and is composed of K exhaustive and exclusive hypotheses $\Theta = \{\omega_1, \omega_2, ..., \omega_K\}$. A BBA $m$ is written as follows:

$$\begin{cases} \sum\limits_{A \subseteq \Theta} m(A) = 1 \\ m(\emptyset) \geq 0. \end{cases} \tag{10.4}$$

Assuming that a source of information has a reliability rate equal to $(1-\alpha)$ where $(0 \leq \alpha \leq 1)$, such a meta-knowledge can be taken into account using the discounting operation introduced by [Sha76a], and is defined by:

$$\begin{cases} m^\alpha(A) = (1-\alpha) \times m(A) & \forall A \subset \Theta \\ m^\alpha(\Theta) = (1-\alpha) \times m(\Theta) + \alpha. \end{cases} \tag{10.5}$$

A discount rate $\alpha$ equal to 1 means that the source is not reliable and the piece of

information that is provided cannot be taken into account. On the contrary, a null discount rate indicates that the source is fully reliable and the piece of information that is provided is entirely acceptable.

Within the evidence theory, several combination rules have been introduced, and this work focuses on the Dempster rule of combination [Dem67]. Assuming two BBAs $m_1$ and $m_2$ modelling two independent reliable sources of information $S_1$ and $S_2$, the Dempster rule of combination is defined as follows:

$$m = m_1 \oplus m_2, \tag{10.6}$$

so that :

$$m = \frac{1}{1-m(\emptyset)} \sum_{B \cap C = A} m_1(B) \times m_2(C) = \frac{1}{1-m(\emptyset)} m(A), \quad \forall A \subseteq \Theta, A \neq \emptyset, \tag{10.7}$$

where $m(\emptyset)$ is defined by:

$$m(\emptyset) = \sum_{B' \cap C' = \emptyset} m_1(B') \times m_2(C') = m(\emptyset). \tag{10.8}$$

$m(\emptyset)$ represents the conflict mass between $m_1$ and $m_2$.

The pignistic probability, denoted $BetP$, is proposed by Smets et al. [Sme05] within the Transferable Belief Model (TBM). In the decision phase, the pignistic transformation consists in distributing equiprobably the mass of a proposition A on its included hypotheses. Formally, the pignistic probability is defined by:

$$BetP(\omega_n) = \sum_{A \subseteq \Theta} \frac{|\omega_n \cap A|}{|A|} \times m(A), \qquad \forall \omega_n \in \Theta. \tag{10.9}$$

where || is the cardinality operator.

### 10.3.2 Evidential *c*-means (ECM)

This subsection presents the ECM clustering algorithm [MD08a]. The ECM algorithm is based on the concept of credal partition, which extends those of fuzzy and possibilistic ones. To derive such a structure, the following objective function is

minimized:

$$J_{ECM}(M,V) \triangleq \sum_{i=1}^{d} \sum_{\{j/A_j \neq \emptyset, A_j \subseteq \Theta\}} c_j^{\alpha} m_{ij}^{\beta} dist_{ij}^2 + \sum_{i=1}^{n} \delta^2 m_{i\emptyset}^{\beta}, \tag{10.10}$$

subject to:

$$\sum_{\{j/A_j \neq \emptyset, A_j \subseteq \Theta\}} m_{ij} + m_{i\emptyset} = 1, \quad \forall i = 1, \ldots, d, \tag{10.11}$$

where $m_{i\emptyset}$ and $m_{ij}$ respectively denote $m_i(\emptyset)$ and $m_i(A_j)$. M is the credal partition $M = (m_1, \ldots, m_d)$ and V is a cluster centers matrix. $c_j^{\alpha}$ is a weighting coefficient and $dist_{ij}$ is the Euclidean distance. In this work, the default values prescribed by the authors in [MD08a] are used, i.e. $\alpha = 1$, $\beta = 2$ and $\delta = 10$. A more detailed introduction to the evidential theory and ECM can be found in Appendix B.

### 10.3.3 The hybrid evidential ontology-based approach for predictive maintenance

This section introduces our proposed hybrid method for failure time and criticality prediction. We follow the different steps introduced in Fig. 10.1. The only difference is the selection of learning method. In the evidential ontology-based approach, the evidential *c*-means algorithm is used for evaluating failure criticality, instead of the fuzzy *c*-means algorithm in Fig. 10.1. The approach starts with the SPM on machine historical data [AS+95]. Then, the ECM algorithm is applied to cluster the failures according to their criticality, based on failure temporal constraints and estimated maintenance cost. After the clustering, different clusters are labeled with criticality *Low*, *Medium*, and *High*. With obtaining the results from ECM, MFPO with its SWRL rule-based extensions are used to predictive the criticality of a future failure.

#### Evidential clustering for failure criticality

In Chapter 9, one ontology-based failure prediction method is proposed. However, the method is based on crisp logic, and it fails to identify the criticality of a failure into the correct category when there are uncertain situations. To cope with this issue, an evidential approach which is able to handle such type of uncertainty situations is required. To do so, the times to failures described in rules and the estimated maintenance cost are used as training examples for ECM with 3 fixed clusters which represents three levels of criticality: i) *high criticality*, which indicates the time from

a normal condition to the failure is relatively short and the production line should be stopped for immediate maintenance, or the estimated maintenance cost is relatively high; ii) *medium criticality*, indicating the failure may happen after a moderate amount of time, or the estimated maintenance cost is moderate; iii) *low criticality*, indicating the failure may happen in the long future and machine operators will have sufficient time to plan maintenance actions, or the estimated maintenance cost is relatively low.

In this subsection, two factors are considered to evaluate the criticality of a failure. Assuming a prediction rule in a form $\mathrm{R}: \mathrm{A} \xrightarrow[[t_i,t_j]]{} \mathrm{F}ailure$ that predicts the failure with a time interval with an Estimated Maintenance Cost EMC. The time to failure and the cost of the failure are valuable descriptors to assess the criticality of a failure. Each rule R has a value of support that evaluates its pertinence. The aim is to use both predicted maintenance cost and predicted temporal constraints of the failure within a rule to assess the criticality of a predicted failure.

Let us assume a sequence S classified by a rule R as a failure in $[t_i, t_j]$ with an EMC. A BBA is computed from both parameters on the frame of discernment $\{\mathrm{L}ow, \mathrm{M}eduim, \mathrm{H}igh\}$ for each level of criticality, based on the ECM algorithm. Both $m_{\mathrm{S},Cost}$ and $m_{\mathrm{S},time}$ are discounted using the support of the used rule R as follows:

$$m_{\mathrm{S}}^{1-\mathrm{S}up(\mathrm{R})} = m_{\mathrm{S},Cost}^{1-\mathrm{S}up(\mathrm{R})} \oplus m_{\mathrm{S},time}^{1-\mathrm{S}up(\mathrm{R})}. \tag{10.12}$$

$m_{\mathrm{S}}^{1-\mathrm{S}up(\mathrm{R})}$ is the BBA obtained from the aggregation of the cost and the time to failure BBAs using the Dempster rule of combination. $1-\mathrm{S}up(\mathrm{R})$ is seen as the reliability value used to discount the obtained BBAs. The final level of criticality is decided upon the use of the arguments of the maxima of the pignistic probability as follows:

$$\mathrm{H}_n = \underset{\omega_n \in \Theta}{\mathrm{argmax}} \mathrm{B}et\mathrm{P}(\omega_n). \tag{10.13}$$

**Ontology reasoning for failure time and criticality prediction**

To predict time and criticality of future failures, SWRL rules are proposed for ontology reasoning. The proposed SWRL rules reason on the individuals in the MFPO ontology (introduced in Chapter 8.5), and infer new knowledge about failure prediction.

As introduced in Section 10.3.3, after the implementation of SPM on data sets, the obtained frequent patterns are in a rule format $\mathrm{A} \xrightarrow[[t_1,t_2]]{} \mathrm{F}ailure$. Within these patterns, the *State (Event)* of a *ManufacturingResource* is determined by a set of *ObservableProperties* (with their associated values). In this context, the antecedent of such a rule is constructed by a conjunction of the time intervals among these events and the quantitative values of *ObservableProperties* (attributes). The consequent of such a rule comprises the temporal constraints of the failure and its criticality.

In this work, the frequent chronicle mining algorithm introduced in [SST18] is used to obtain chronicles, which extract a special type of sequential patterns in a rule format. After that, SWRL rules are proposed to formalize the mining results and to predict failures. To enable the generation of SWRL rules, a novel algorithm is proposed to transform chronicles into SWRL predictive rules. The pseudo-code of the rule transformation algorithm is shown in Algorithm 3. Similar to previous algorithms, the $\wedge$ symbol means the conjunction of rule atoms. The algorithm is executed in four major steps:

1) The function *LastNonfailureState* extracts the last non-failure state (event) within a chronicle, and the function *LastNonfailureState* extracts the failure event within a chronicle.

2) For each time interval in a chronicle, the two functions *PrecedingEvent* and *SubsequentEvent* extract the preceding and subsequent events of it. Then the two events and this time interval forms different atoms in the antecedent of the rule, and they are treated as conjunctions. After that, the algorithm extracts the time interval and its duration between the last non-failure state and the failure, to construct the atoms in the consequent. The constructed atoms are also treated as conjunctions.

3) The ECM algorithm is applied to assess the failures according to their criticality. The failures are assigned to three categories, and three object properties in MFPO are used to represent the pignistic probability to different clusters. The pignistic probabilities are treated as a conjunction, to form the consequent of the rule.

4) At last, a rule is constructed as an implication between the antecedent and the consequent.

**Algorithm 3** Algorithm to transform a chronicle into a predictive SWRL rule, based on evidential *c*-means (ECM) algorithm

**Input:** $\mathscr{S}_{\mathrm{F}}$: a chronicle within which the last state (event) is a failure, $\mathscr{E}$: a set of the states that are described within a chronicle.

**Output:** R ▷ R: the SWRL rule to be constructed.

1: $ls \leftarrow \mathrm{L}ast\mathrm{N}onfailure\mathrm{S}tate(\mathscr{S}_{\mathrm{F}}, \mathscr{E})$ ▷ Extract the last non-failure state before the failure within a chronicle.

2: $f \leftarrow the\mathrm{F}ailure(\mathscr{E})$ ▷ Extract the failure within a chronicle.

3: $\mathrm{R} \leftarrow \emptyset$, $\mathrm{A} \leftarrow \emptyset$, $\mathrm{C} \leftarrow \emptyset$, $\mathrm{A}tom_a \leftarrow \emptyset$, $\mathrm{A}tom_c \leftarrow \emptyset$, $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{L}ow} = 0$, $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{M}edium} = 0$, $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{H}igh} = 0$. ▷ A: the antecedent of the SWRL rule. C: the consequent of the SWRL rule. $\mathrm{A}tom_a$: a subset of all atoms within the antecedent. $\mathrm{A}tom_c$: a subset of all atoms within the consequent. $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{L}ow}$, $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{M}edium}$, $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{H}igh}$: the three pignistic probabilities to the three categories of failure criticality.

4: **for each** $e_i \in \mathscr{E}$ **do**

5: $pe \leftarrow \mathrm{P}receding\mathrm{S}tate(e_i, \mathscr{S}_{\mathrm{F}})$ ▷ Extract the preceding state

6: $se \leftarrow \mathrm{S}ubsequent\mathrm{S}tate(e_i, \mathscr{S}_{\mathrm{F}})$ ▷ Extract the subsequent state

7: $\mathrm{A}tom_a \leftarrow pe \wedge se$

8: $\mathrm{A} \leftarrow \mathrm{A}tom_a \wedge \mathrm{A}$

9: **end for**

10: $ftd \leftarrow \mathrm{F}ailure\mathrm{T}ime\mathrm{D}uration(ls, f)$ ▷ Extract the time duration between the last non-failure state and the failure.

11: $mc \leftarrow \mathrm{M}aintenance\mathrm{C}ost(\mathscr{S}_{\mathrm{F}})$ ▷ Obtain the estimated maintenance cost for the failure described in this the chronicle.

12: $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{L}ow} \leftarrow \mathrm{P}ignistic\mathrm{P}robability\mathrm{L}ow(ftd, mc)$

13: $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{M}edium} \leftarrow \mathrm{P}ignistic\mathrm{P}robability\mathrm{M}edium(ftd, mc)$

14: $\mathrm{F}_{Failure\mathrm{C}riticality\mathrm{H}igh} \leftarrow \mathrm{P}ignistic\mathrm{P}robability\mathrm{H}igh(ftd, mc)$ ▷ Use the ECM algorithm to compute the pignistic probabilities of this criticality to the three clusters.

15: $\mathrm{C} \leftarrow \mathrm{F}_{Failure\mathrm{C}riticality\mathrm{L}ow} \wedge \mathrm{F}_{Failure\mathrm{C}riticality\mathrm{M}edium} \wedge \mathrm{F}_{Failure\mathrm{C}riticality\mathrm{H}igh} \wedge ftd$

16: $\mathrm{R} \leftarrow (\mathrm{A} \rightarrow \mathrm{C})$ ▷ A rule is generated as an implication between the antecedent and consequent.

17: **return** R

## 10.4 Experimentation and results evaluation

### 10.4.1 Experimental results for fuzzy clustering

The two hybrid ontology-based approaches are applied to the SECOM data set. To improve the quality of failure prediction and criticality assessment, *Chronicle Support* is chosen as a reference measure, to select the most relevant failure chronicles. As a result, the same set of chronicles are extracted from the SECOM data set, which is shown in Table 10.2.

After applying the FCM algorithm to the numeric values of the minimum time duration ($Min_{\mathrm{TD}}$) among the last normal events and the failures, the degrees of membership of failure criticality to different categories are obtained. Table 10.1 gives the clustering results. For each failure chronicle, the assessment of failure criticality is associated with a fuzzy index, indicating the degree of its membership to Criticality High (CH) cluster, Criticality Medium (CM) cluster, and Criticality Low (CL) cluster.

Table 10.1 – The failure criticality assessment results of the 10 failure chronicles.

| Failure Chronicle | $Min_{\mathrm{TD}}$ (millisecond) | CH | CM | CL |
|---|---|---|---|---|
| $\mathscr{C}_{\mathrm{F1}}$ | 1020 | 0.988364 | 0.002024 | 0.009611 |
| $\mathscr{C}_{\mathrm{F2}}$ | 1560 | 0.998590 | 0.001052 | 0.000358 |
| $\mathscr{C}_{\mathrm{F3}}$ | 2280 | 0.998489 | 0.000231 | 0.001280 |
| $\mathscr{C}_{\mathrm{F4}}$ | 4320 | 0.588961 | 0.216887 | 0.194152 |
| $\mathscr{C}_{\mathrm{F5}}$ | 7440 | 0.256704 | 0.035270 | 0.708026 |
| $\mathscr{C}_{\mathrm{F6}}$ | 10500 | 0.069900 | 0.197297 | 0.732802 |
| $\mathscr{C}_{\mathrm{F7}}$ | 12840 | 0.056136 | 0.111810 | 0.818294 |
| $\mathscr{C}_{\mathrm{F8}}$ | 13800 | 0.034679 | 0.111810 | 0.832054 |
| $\mathscr{C}_{\mathrm{F9}}$ | 18640 | 0.029900 | 0.046937 | 0.918384 |
| $\mathscr{C}_{\mathrm{F10}}$ | 22380 | 0.000099 | 0.000303 | 0.999598 |

Based on Algorithm 2 and the failure assessment results in Table 10.1, the failure chronicles were transformed to SWRL rules. Fig. 10.2 presents an example SWRL rule that is obtained after implementing the whole process. Firstly, one frequent chronicle that contains a failure is extracted by applying the frequent chronicle mining approach. Secondly, after the mining of frequent failure chronicles, the numeric value of the $Min_{\mathrm{TD}}$ between the last normal state (event) inside a failure chronicle and the failure is clustered by the FCM algorithm. The arrow at the right side of the figure refers to the data point in the FCM clustering results that repre-

sents the $Min_{\mathrm{TD}}$ value contained in the failure chronicle. Thirdly, one SWRL rule is constructed based on the degrees of membership to the three categories. In this rule, *hasA63V*, *hasA102V*, *hasA209V*, *hasA347V*, *hasA476V* are data properties in the MFPO ontology that link individuals of the *State* class with XML Schema Datatype values. They correspond to the quantitative values of the attributes $A_{63}$, $A_{102}$, $A_{209}$, $A_{347}$, and $A_{476}$ in the SECOM data set. *TimeInterval* corresponds to the root class of all individuals of time intervals. There are two object properties that link *TimeInterval* with *State*: *hasSubState* and *hasProState*, among which *hasSubState* corresponds to the subsequent state of a time interval, and *hasProState* indicates the preceding state of a time interval. In this case, state $S_1$ is the preceding state of the time interval between $S_1$ and $S_2$, and state $S_2$ is the subsequent state of this time interval. To describe the numerical intervals which are obtained by discretization, SWRL Built-Ins are used to specify the upper and lower numerical boundaries. The consequent of this rule comprises time constraints of the failure, different categories of failure criticality, and their degrees of membership. The minimum and maximum time duration between a state with the failure is described by the data property *hasMinF* and *hasMaxF*. The degrees of membership to different criticality categories of the failure is given by the FCM algorithm, and they are computed based on the minimum time duration between a normal state to the failure. By this way, the assessment of failure criticality is inferred by the launching of such a predictive SWRL rule.

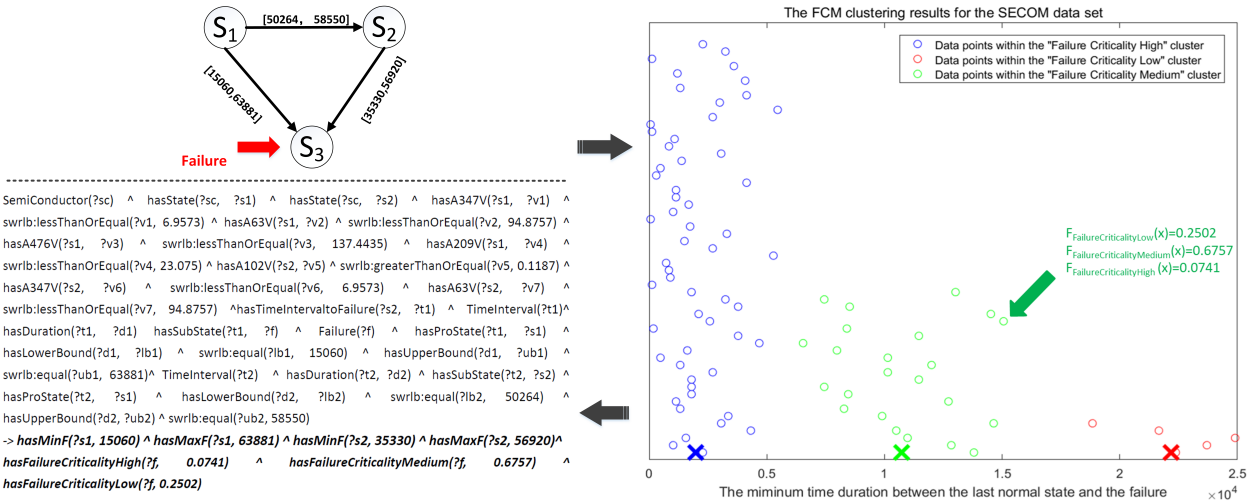



Figure 10.2 – The procedure of transforming a failure chronicle into a SWRL rule, based on the failure criticality assessment results.

Following this procedure, the failure criticality assessment presented in Table.

10.1 are formalized by the MFPO ontology and SWRL rules, which aim to facilitate the predictive maintenance of the semi-conductor manufacturing process.

### 10.4.2 Experimental results for evidential clustering

The evidential clustering approach is validated on several synthetic data sets and the SECOM data set. The experimentation starts with the preprocessing of data, followed by the chronicle mining step. Similar to the fuzzy clustering approach, the frequent chronicle mining algorithm introduced in [SMS+19] is used to extract frequent chronicles.

#### Experimentation on synthetic data sets

After obtaining the chronicles, synthetic data is generated for the estimated maintenance cost. To do this, the maintenance cost is generated as uniformly distributed random numbers between [0,100]. In the generated data, each value of maintenance cost is associated with a failure, indicating the estimated maintenance cost caused by the failure. In addition to the temporal constraints of failures, maintenance cost is considered as the second descriptor for the failure criticality. The third step is to apply ECM on the synthetic data set, for determining the criticality of failures based on their temporal constraints and estimated maintenance cost. Following the evidential clustering approach introduced in Chapter 10.3.3, the final levels of criticality of the failures are obtainted. At last, the extracted frequent chronicles are transformed into SWRL predictive rules (using Algorithm 3), and the ECM clustering results are also formalized by these rules.

Table 10.2 shows the 10 failure chronicles (FC) which have the highest chronicle support (CS) among all the extracted ones. In this figure, the numeric values of the minimum time duration ($Min_{\mathrm{TD}}$, time unit: second) among the last normal events and the failures, the EMC for each chronicle, and the pignistic probability of the final criticality (PPFC) are presented. For the clustering results, the final level of a failure's criticality is shown inside the brackets within the last column of the table.

#### The Generation of SWRL Rules Based on Chronicles and ECM Results

To formalize the failure criticality assessment results and to predict the criticality of future failures, SWRL rules are generated based on the obtained chronicles and ECM clustering results. To do this, Algorithm 3 is used to transform the failure chronicles

Table 10.2 – Failure chronicles that have the 10 highest chronicle support, and their failure criticality assessment results.

| $\mathscr{C}_F$ | $Min_{TD}$ | EMC | CS | PPFC |
|---|---|---|---|---|
| $\mathscr{C}_{F1}$ | 10 | 33.4163 | 96.19% | 0.6652 (Medium) |
| $\mathscr{C}_{F2}$ | 7 | 50.0472 | 95.61% | 0.5049 (Medium) |
| $\mathscr{C}_{F3}$ | 3 | 14.9865 | 94.48% | 0.6140 (High) |
| $\mathscr{C}_{F4}$ | 4 | 17.3388 | 94.21% | 0.8739 (Medium) |
| $\mathscr{C}_{F5}$ | 21 | 81.8148 | 92.94% | 0.3921 (Low) |
| $\mathscr{C}_{F6}$ | 3 | 65.9605 | 91.06% | 0.8796 (High) |
| $\mathscr{C}_{F7}$ | 11 | 68.1971 | 90.27% | 0.4722 (Medium) |
| $\mathscr{C}_{F8}$ | 24 | 9.6730 | 90.01% | 0.6871 (Low) |
| $\mathscr{C}_{F9}$ | 10 | 64.8991 | 86.93% | 0.4266 (Medium) |
| $\mathscr{C}_{F10}$ | 18 | 66.6338 | 86.87% | 0.4030 (Low) |

and ECM clustering results into predictive SWRL rules. Fig. 10.3 presents an example SWRL rule that is generated following our approach. Within this rule, the pignistic probability of the final criticality category of the failure is given by the ECM algorithm. In this way, the assessment of the failure criticality is inferred by the launching of such a predictive SWRL rule.

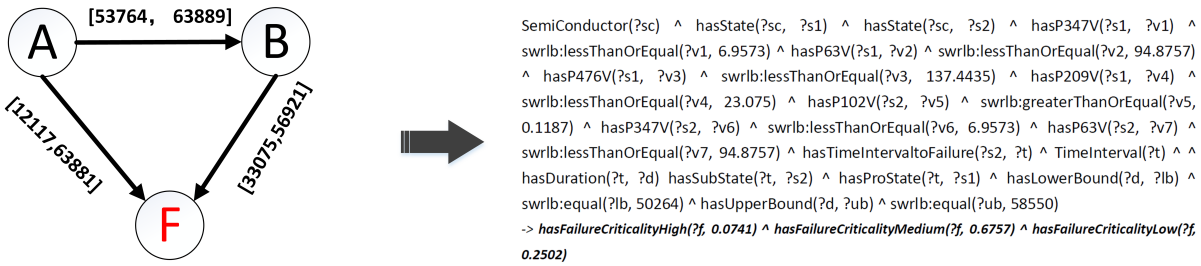



Figure 10.3 – An example SWRL rule generated from a chronicle.

To evaluate the quality of the SWRL rules, two measures are computed. We identify *Accuracy* and *Coverage* as the two rule measures for evaluating the quality of SWRL rules, as these two measures are the important indicators of a rule's reliability [AC99]. These two measures can be derived based on the analysis of a $2 \times 2$ contingency table. A contingency table for a rule $R : A \rightarrow F$ is computed as follows:

In Table 10.3, $n_{af}$ denotes the number of training examples for which both the antecedent and the consequent of the rule are true. $n_{a\bar{f}}$ denotes the number of training examples for which the antecedent of the rule is true, and the consequent of the rule is not true. On the other hand, $n_{\bar{a}f}$ denotes the number of training examples

Table 10.3 – The contingency table for computing rule quality measures.

| | F **is true** | F **is not true** | |
|---|---|---|---|
| A is true | $n_{af}$ | $n_{a\bar{f}}$ | $n_a$ |
| A is not true | $n_{\bar{a}f}$ | $n_{\bar{a}\bar{f}}$ | $n_{\bar{a}}$ |
| | $n_f$ | $n_{\bar{f}}$ | N |

for which the antecedent is not true, and the consequent of the rule is true. $n_{\bar{a}\bar{f}}$ denotes the number of training examples for which neither the antecedent and the consequent of the rule is true. $n_a$, $n_{\bar{a}}$, $n_f$, $n_{\bar{f}}$ are marginal totals, and N is the total number of training examples.

Based on the information provided by the contingency table, two basic evaluation measures can be computed: *Accuracy* and *Coverage*. Among them, the rule accuracy is defined as

$$Accuracy(\mathrm{R}) = n_{af}/n_a, \tag{10.14}$$

and rule coverage is defined as

$$Coverage(\mathrm{R}) = n_{af}/n_f. \tag{10.15}$$

The above two equations are used to obtain the average value of *Accuracy* and *Coverage* for the SWRL rules. Table 10.4 presents the two measures under different chronicle support. It can be observed from the table that as the chronicle support increases, the accuracy of rules also increases. It is reasonable as the support of extracted chronicles increases, more relevant chronicles are extracted. On the other hand, as the number of extracted rules decreases, the sequences that are covered by the rules decreases. This is the reason why the average value of coverage shows a downtrend.

Table 10.4 – Two rule quality measures under different chronicle support.

| Chronicle support | Accuracy | Coverage |
|---|---|---|
| 0.5 | 76.52% | 74.26% |
| 0.6 | 74.14 % | 75.71% |
| 0.7 | 76.98 % | 74.35% |
| 0.8 | 79.33% | 70.49% |
| 0.9 | 82.56% | 68.10% |
| 1 | 84.45% | 67.71% |

**Experimentation on a real-world data set**

To evaluate the performance of the prediction and failure criticality assessment, ECM is applied on the SECOM data set [MJ08], which contains measurements of features of semi-conductor productions within a semi-conductor manufacturing process.

As a result, the hard credal partition on the SECOM data set is computed. In total, at most $2^{\Theta}$ focal sets could be obtained through credal partition, where $\Theta$ is the frame of discernment. In our experimentation, $\Theta$ represents the three levels of failure criticality. For the SECOM data set, the temporal constraints of failures are considered as the descriptor for criticality. The data points on the empty set which have the highest masses are removed as outliers before they are assigned to the clusters. Since cluster $\omega_l$ is labeled as *low criticality*, $\omega_m$ as *medium criticality*, and $\omega_h$ as *high criticality*, there is no hesitation between the clusters $\omega_l$ and $\omega_h$. At last, since the outliers are removed before the credal partition, $2^3 - 2 = 6$ focal sets are obtained.

Fig. 10.4 shows the hard credal partition computed on the SECOM data set with the following parameters: $\alpha = 1$, $\beta = 2$, $\delta = 10$, and $\varepsilon = 10^{-3}$. The X axis stands for the temporal constraints of the failures, Y axis indicates the number of training examples that have been assigned to different clusters. As results, 6 focal elements are obtained, including the universal set $\Theta_\omega = \{\omega_l, \omega_{lm}, \omega_m, \omega_{mh}, \omega_h\}$. Among them, $\omega_l$ is the focal set representing the *low criticality* class, $\omega_m$ is the focal set representing the *medium criticality* class, and $\omega_h$ is the focal set representing the *high criticality* class. $\omega_{lm}$ is the hesitation between the $\omega_l$ and $\omega_m$ clusters, which is $\{\omega_l, \omega_m\}$. $\omega_{mh}$ is the hesitation between the $\omega_m$ and $\omega_h$ clusters, which means $\{\omega_m, \omega_h\}$. The center of each class is marked as a cross.

It can be observed that the $\omega_h$ class has the highest number of training examples, and over half of the failures are assigned to the $\omega_h$ and $\omega_{mh}$ clusters. As the value of a temporal constraint increases, the criticality level of the failure decreases. It also can be seen that the evidential-based clustering extends the fuzzy and possibilistic methods by not only assigning data points to single clusters but also to all subsets of the universal set $\Theta_\omega$. In this way, ECM provides more insights into failures than classical clustering methods. This advantage of ECM facilitates the reasoning about uncertainty that may exists within the overlapping among different clusters.

To obtain the final level of criticality, the pignostic probability B$et$P and the maxima of B$et$P are computed. After comparing the B$et$P of the three clusters, the class

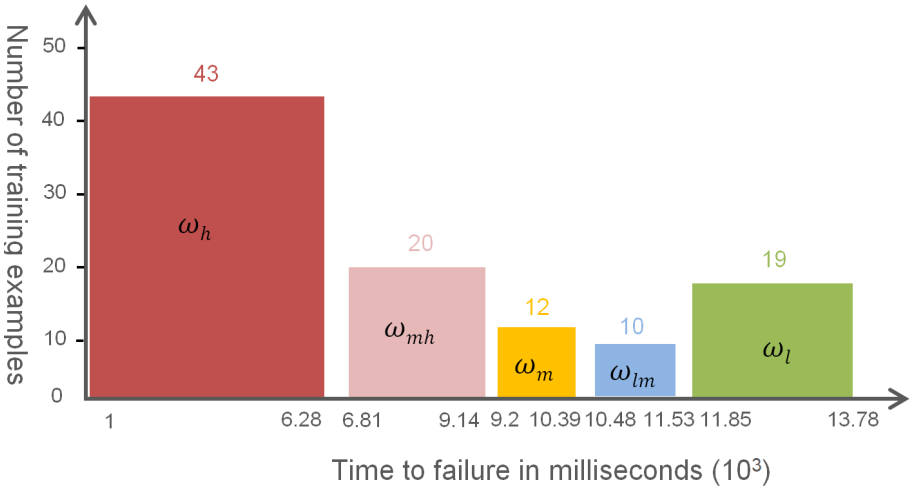



Figure 10.4 – Hard credal partition for the SECOM data set.

with the maximum B$et$P is selected to represent the final level of criticality. Fig. 10.5 shows the final criticality for the training examples in the SECOM data set. $\omega_l$, $\omega_m$, $\omega_h$ represents the *low criticality* class, *medium criticality* class, and *high criticality* class respectively. It can be seen that there is no hesitation among different clusters, which ensures the final criticality to be determined based on a maximum of B$et$P of the three clusters.

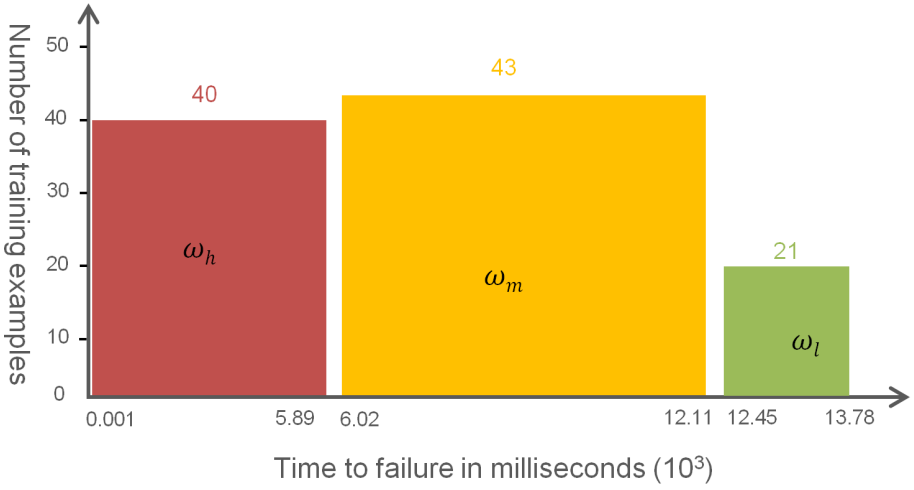



Figure 10.5 – Final criticality levels of failures in the SECOM data set.

An example of the ECM clustering results on the training data is shown in Table 10.5. Within this example, rule #45 is selected to show the obtained BBAs and the pignistic probability of the final criticality (PPFC) of the failure which is described within this rule. Since the *high criticality* class is assigned with the highest PPFC,

the final decision on the criticality level of this failure is *high*.

Table 10.5 – Experimental results of a training example in the SECOM data set.

| Failure index | BBAs of the failure | PPFC |
|---|---|---|
| #45 | $m(\{\omega_h\}) = 0.3174$<br>$m(\{\omega_{mh}\}) = 0.2219$<br>$m(\{\omega_m\}) = 0.2929$<br>$m(\{\omega_{lm}\}) = 0.0181$<br>$m(\{\omega_l\}) = 0.0485$<br>$m(\{\Theta_\omega\}) = 0.1012$ | $BetP(\omega_h) = 0.4683$<br>$BetP(\omega_m) = 0.4391$<br>$BetP(\omega_l) = 0.0926$ |

## 10.5 Summary

In this chapter, we tackle the assessment of failure criticality levels by introducing two hybrid ontology-based approaches. The proposed approaches are based on the combined use of clustering techniques and semantics, where fuzzy and evidential clustering techniques are used to learn the criticality of failures based on machine historical data, and semantic technologies use the results of clustering to predict the time of failures and the criticality of them. The contributions of this chapter lie firstly in the formalization of the predictive maintenance knowledge based on ontologies, by which the SPM results are formalized and the criticality of failures are inferred. Secondly, the assessment of failure criticality levels is achieved by implementing an unsupervised learning approach. The unsupervised learning approach uses FCM and ECM algorithms to cluster failures according to their time of occurrences, which reflects the criticality of them. After that, each cluster is labeled with a criticality index (high, medium, or low) to assess failure criticality according to their temporal constraints and estimated maintenance cost. Thirdly, the SWRL rules are proposed to formalize the assessment results, in order to enable ontology reasoning for predictive maintenance.

As future work, we aim to consider more failure descriptors for criticality levels. Also, we plan to study the impact of different parameter values on the final criticality within the two clustering algorithms. This will allow us to choose more flexible scenario for assessing the failure criticality according to different real-world contexts.

# Chapter 11

# Using rule quality measures for rule base pruning and integration

## Contents

## 11.1 Introduction

Within a knowledge-based system, a rule describes how axioms (domain knowledge) can be combined to derive new information [Ste85]. It is a conditional statement that links given conditions to actions or outcomes. Essentially, rules are *if-then* statements, which are used to formulate the conditional statements that comprise the knowledge base [Abr05]. In the domain of predictive maintenance, rule-based condition monitoring and diagnostic systems are employed to detect incipient anomalies. Within such a system, decision rules are used to represent the relations among real-world machinery conditions with each possible anomaly, such as machinery fault and failure.

As introduced in Chapter 9 and 10, frequent chronicle mining is a promising technique for predicting not only the order of the non-failure events but also the temporal intervals among them. The output of frequent chronicle mining is a set of failure chronicles that are in the form of logical rules. This type of rules (in this chapter, they are defined as *chronicle rules*) describes different events as well as their temporal constraints and predicts the time duration within which the future failures occur.

Normally, the number of chronicle rules extracted from frequent chronicle mining is large. Due to a certain degree of imprecision in real-world data, some of the extracted chronicle rules may suffer from low-quality. As a result, low-quality chronicle rules (e.g., rules with low accuracy in failure prediction) may reduce the efficiency and accuracy of rule-based reasoning. Therefore, a rule base pruning method for obtaining a high-quality subset of rules is needed.

On the other hand, as industrial big data are collected from a variety of devices and environments, the decision rules that are extracted from them may be obtained from heterogeneous sources. This may cause a set of problems (e.g., rule contradiction, subsumption, and redundancy) when heterogeneous rules are combined to achieve a satisfactory level of reasoning performance. Using a rule base of low quality may result in inappropriate maintenance decisions, thus weakening the performance of knowledge-based maintenance systems.

In this chapter, a novel rule base refinement approach is proposed. The proposed approach consists of two rule refinement methods that aim to progressively improve the quality of a rule base:

- A rule pruning/reduction method applied to the rule base obtained from fre-

quent chronicle mining. To reduce the number of extracted rules and to achieve the best quality of the rule base, a multi-objective optimization approach is applied. The approach aims to maximize rule *Accuracy* and *Coverage* for obtaining a set of rules with the best quality.

- A rule integration method for combining chronicle rules and expert rules. To enhance the performance of failure prediction, experts' experience needs to be capitalized in the form of expert rules when the chronicle rules fail to provide correct decisions. In this work, we consider the expert rules that have a similar structure as chronicles. When these expert rules are integrated with the chronicle rules, they may suffer from problems such as *Redundancy*, *Contradiction*, and *Subsumption*. In this context, a rule integration method for detecting the three problems is proposed.

## 11.2 Related work and existing challenges

Given the importance of rule quality measures for the refinement of a rule base, a number of related research works have been proposed. In this section, a literature review is presented with focus on those contributions that use rule quality measures to guide the pruning and integration processes of rule bases.

The first work to mention is proposed by An and Cercone, which formalizes rule quality measures in the form of statistical and empirical formulas [AC99]. In their work, the authors perform a comparison among different rule quality formulas by evaluating the predictive accuracy upon several standard data sets. To do this, a set of rule quality measures is applied to 27 data sets. Moreover, a meta-learning method is introduced to discover the relationships among the characteristics of data sets and rule quality formulas. As a result, meta-rules that represent the learned relationships are proposed to guide the selection of rule quality formulas before inducing rules from data sets. In the same topic, Marek Sikora pays special attention to two rule quality measures: *Accuracy* and *Coverage* [Sik06]. Based on these two measures, a rule ranking method is presented to select a subset of rules that are characterized by good generalization capabilities. Experimental results show that their method can be used in the rule induction and reduction processes. However, the rules are ranked according to different quality measures separately, thus lacking the capability of considering multiple quality measures at the same time. In another

work of Sikora and Wróbel, a data-driven adaptive selection method for rule quality measures is introduced [SW11]. To determine and select the appropriate measures, an evaluation of the quality of the qualifier is performed. After comparing the performance of the adaptive selection method with the arbitrary selection method, the authors concluded that their method shows statistically better results that the arbitrary ones.

The evaluation and refinement of rules bases are also studied as in the optimization research field. Normally, this type of methods treats the selection of rules with specific properties as a multi-objective optimization problem. The objectives to be optimized are several rule quality measures such as *Accuracy*, *Coverage*, *Surprisingness*, and *Logical Sufficiency*. Within this topic, the first work to be mentioned is proposed in [DLIPBRS03] and [dlIRRS05]. In their works, multi-objective optimization evolutionary algorithms are used to select several interesting measures and to propose an approximation to the Pareto-optimal set according to those measures. The authors use rule *Accuracy* and *Coverage* as interesting measures and implement the Fast Elitist Non-Dominated Sorting Genetic Algorithm (NSGA) to derive the Pareto-optimal set. To improve the quality of obtained rule sets, they combine the use of the crowding mechanism in NSGA-II and the innovative approaches to cluster rules according to dissimilarity measures. The results have shown that the rule sets obtained by NSGA-II in the standard implementation do not contain many cases of rules that are close in the objective space but far apart in terms of their support sets [dlIRRS05]. To overcome this issue, the authors introduce the concept of rule dissimilarity in the crowding measure in NSGA-II. This allows to increase the diversity of obtained rules. Following this work, Ishibuchi and Namba propose a three-stage rule extraction method based on data [IN04]. In their method, the number of selected rules and the length of a rule are selected as the optimization criterion. The experimental results show the method is applicable to high-dimensional pattern classification problems with many continuous attributes. The main characteristic feature of the their approach is that many rule sets with different accuracy and different complexity are simultaneously obtained from its single run [IN04]. Martin et al. extend the NSGA and NSGA-II to perform evolutionary learning of the intervals of the attributes and a condition selection for each rule [MRAFH14]. *Comprehensibility*, *Interestingness*, and *Performance* are selected as three objectives to optimize the performance of quantitative association rules. The algorithm is compared with other mono-objective and multi-objective algorithms. Results show the rules ob-

tained from the proposed algorithm have higher interesting measure and higher coverage.

The review of the existing research works posts two main challenges. Most of the existing research works analyze rule quality measures according to the contingency table. These research works lack the capability of resolving other problems such as rule *Redundancy*, *Contradiction*, and *Subsumption*. Moreover, most of the rule quality analysis and rule base refinement methods focus on a single source of rule extraction (e.g., association rule induction from a single data set). These methods may fail to address rule the three issues when decision rules are obtained from heterogeneous sources and in various formats. Therefore, a rule base refinement method that is not only capable of pruning a single rule base but also detecting the three issues among heterogeneous rules should be proposed.

## 11.3 Rule quality measures for chronicle rules

To introduce the rule quality measures for pruning a chronicle rule base, we use rule *Accuracy* and *Coverage* as two important measures for evaluating the reliability of chronicle rules. The definition of these two measures are introduced in Chapter 10.4.2.

Since the data set from where the chronicles are mined may not cover all real-world conditions, it is conceivable that the chronicle rules may fail to predict a failure. In this case, it is required to consult domain experts for decisions about failure prediction and maintenance. In this way, domain experts use their experience to assess the current state of the system and provide appropriate decisions, in the form of expert rules. By this, when there is a similar situation needs to be addressed, the rule which capitalizes domain experts' experience will be launched together with the initial rules to identify potential failures and to make predictions. This strategy may enrich the *Experience* component of the KREM architecture, which performs as a complementary model to deal with incomplete knowledge models in the *Knowledge* component.

However, when the expert rules are integrated into the initial rule base, a set of issues such as rule redundancy, conflict and subsumption may occur, thus reducing the reliability and performance of the rule-based reasoning process. Therefore, a rule base verification step is required to check any possible issues within an integrated rule base. In this chapter, *Redundancy*, *Conflict*, and *Subsumption* are con-

sidered as three important issues for the integration of the chronicle rule base and the expert rules. Based on the definition of chronicle rules in Definition 8, the formal definition of these three issues are given as follows:

**Definition 9** (Chronicle rule redundancy)**.** *A chronicle rule* $R : EC \wedge TEC \rightarrow TF$ *and a rule* $R' : EC' \wedge TEC' \rightarrow TF'$ *are considered as redundant when the two rules have the same sets of events, and the events have the same temporal constraints, denoted as*

$$ChroRedundancy(R, R') \Longleftrightarrow (EC = EC') \wedge (TEC = TEC') \wedge (TF = TF'). \quad (11.1)$$

**Definition 10** (Chronicle rule conflict)**.** *A chronicle rule* $R : EC \wedge TEC \rightarrow TF$ *and a rule* $R' : EC' \wedge TEC' \rightarrow TF'$ *are considered to have conflict when the two rules succeed in the same sets of non-failure events and the same temporal constraints for these non-failure events, but with conflicting temporal constraints of a failure, denoted as*

$$ChroConflict(R, R') \Longleftrightarrow (EC = EC') \wedge (TEC = TEC') \wedge (TF \neq TF'). \quad (11.2)$$

**Definition 11** (Chronicle rule subsumption)**.** *A chronicle rule* $R : EC \wedge TEC \rightarrow TF$ *subsums the chronicle rule* $R' : EC' \wedge TEC' \rightarrow TF'$ *when they have the same temporal constraints of a failure, but rule* R *contains additional restrictions on the set of non-failure events or on the temporal constraints of these non-failure events. This subsumption relationship is denoted as*

$$ChroSubsums(R, R') \Longleftrightarrow (TF = TF') \wedge ((EC' \wedge TEC') \models (EC \wedge TEC)). \quad (11.3)$$

These definitions are used to detect issues with regard to rule base verification. They provide foundations for the rule base refinement approach.

Fig. 11.1 shows the different steps of the whole approach proposed in this chapter. The whole process starts with the frequent chronicle mining applied to industrial data sets. After that, SWRL rules are generated from the chronicles using algorithm 1. Then the chronicle rule base is pruned to select the best quality rules based on their *Accuracy* and *Coverage*.

In case the selected chronicle rule base fails to make a prediction, the experience capitalization step is triggered to capture experts' experience in the form of

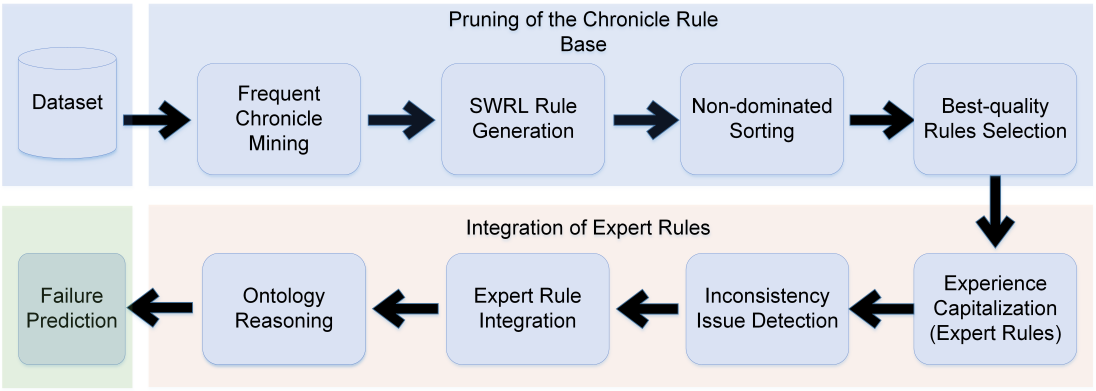



Figure 11.1 – Different steps of the rule base refinement approach.

expert rules. After that, rule *Redundancy*, *Conflict*, and *Subsumption* issues are examined among the chronicle rules and expert rules. If there is no issue, the expert rules are integrated into the chronicle rule base for failure prediction. Otherwise, the integrated rule base is refined according to different situations. The refinement may include discarding the expert rule, or replacing a chronicle rule with an expert rule. By this, the machinery failures are predicted by a refined rule base within a knowledge-based predictive maintenance system.

## 11.4 The multi-objective optimization approach for pruning chronicle rules

To select the set of best quality rules, a multi-objective optimization approach is proposed to prune the chronicle rule base. To do this, we aim to maximize rule *Accuracy* and *Coverage*. The parameters used by the multi-objective optimization algorithm are shown in Table 11.1.

As the aim is to maximize both rule *Accuracy* and *Coverage*, the first objective function is defined as

$$f_1(\mathrm{R}_i) = -\mathrm{A}ccuracy(\mathrm{R}_i) = -n_{af}/n_a. \tag{11.4}$$

The second objective function is computed as

$$f_2(\mathrm{R}_i) = -\mathrm{C}overage(\mathrm{R}_i) = -n_{af}/n_f. \tag{11.5}$$

With obtaining the above two objective functions, a mathematical model for

Table 11.1 – Parameters of the proposed multi-objective optimization approach.

| **Parameter** | **Definition** |
|---|---|
| R | A chronicle rule |
| $p$ | The number of individuals (chronicle rules) |
| $v$ | The number of objective functions |
| $\mathbb{R} = \{R_1, R_2, ..., R_i\}$ | A chronicle rule base of $i$ rules |
| F | The front matrix that stores the individuals at each Pareto front |
| T | The column vector with the rank of each individual |
| $TEC_R$ | The temporal constraints of the non-failure events that are described in R |
| $TF_R$ | The temporal constraints of the failure that are described in R |
| $Accuracy(R)$ | The accuracy of R |
| $Coverage(R)$ | The coverage of R |

chronicle rule base pruning is proposed. The model is shown in Table 11.2. Based on the proposed model, the goal of finding a set of best-quality chronicle rules becomes the search for the set of Pareto-optimal solutions. In this work, the fast non-dominated sorting approach introduced in [OFGC17] is used to obtain the ranking of Pareto dominance for chronicle rules. The pseudocode of this algorithm is shown in Algorithm 4 (A more detailed description of the fast non-dominated sorting procedure is given in Appendix C). After that, the rules within the first front are selected to perform rule-based reasoning. In this way, a set of chronicle rules that have the best quality are launched for failure prediction.

Table 11.2 – The mathematical model for chronicle rule base pruning.

Objective function: $f_1(R_i) = -Accuracy(R_i) = -n_{af}/n_a$
Objective function: $f_2(R_i) = -Coverage(R_i) = -n_{af}/n_f$
Minimize $(f_1(R_i))$
Minimize $(f_2(R_i))$
subject to
$i \leqslant |\mathbb{R}|$
$f_1(R_i) \in [-1, 0]$
$f_2(R_i) \in [-1, 0]$

Fig. 11.2 shows the process of the chronicle rule pruning approach for the chronicle rule base. Firstly, the rule *Accuracy* and *Coverage* measures are computed for each run of the chronicle mining algorithm, under different thresholds of chronicle support. The process stops when the number of mined chronicles stays unchanged. After that, the fitness value of each chronicle rule R is computed. The fitness function is introduced in [SSS82], which is described by the equation:

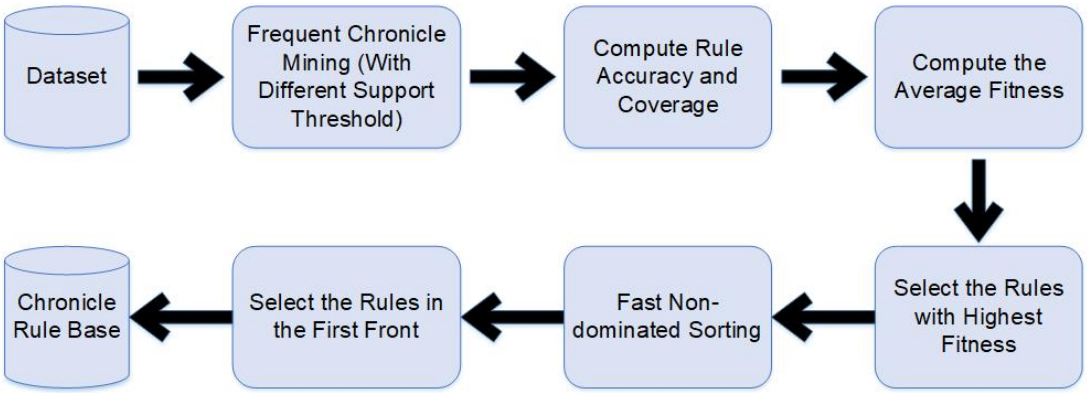



Figure 11.2 – Different steps within the rule base pruning process.

$$Fitness(R) = Accuracy(R) \times Coverage(R). \tag{11.6}$$

The fitness value of a chronicle rule indicates the reliability of it. Rules that have high *coverage* may cover a large number of positive examples. However, these rules may cover negative examples at the same time, which causes the low *accuracy* of them. On the other hand, only considering high-*Accuracy* rules may result too few rules to be selected. This may lead to poor predictive performance of the rules since they may overfit the data. Therefore, the fitness function described above aims to consider these two quality measures at the same time for selecting a set of best-quality rules.

**Algorithm 4** The fast non-dominated sorting procedure, adopted from [OFGC17].

**Input:**

$n$: the dimension of the search space.
$p$: the number of individuals (chronicle rules).
$v$: the number of objective functions.
$\mathrm{P}_{(p\times(n+v+1))}$: population.
$\#\mathrm{M}_i$: the column vector with the number of dominators ($i$) of each individual.
$\#\mathrm{D}_i$: the column vector that contains the element $i$.

**Output:**

F: the front matrix.
T: the rank of each individual.
**Phase 1: Unidirectional Dominance Comparison (UDC)**
**for** $i \leftarrow 1$ **to** $p$ **do**
    $\mathrm{D}_i \leftarrow \{\emptyset\}$
    $\#\mathrm{M}_i \leftarrow 0$
    **for** $j \leftarrow 1$ **to** $p$ **do**
        **for** $k \leftarrow 1$ **to** $v$ **do**
        ▷ Check dominance between individuals $\mathrm{P}_i$ and $\mathrm{P}_j$ for the objective $k$.
        **if** $\mathrm{P}_i \succ \mathrm{P}_j$ **then**
            $\mathrm{D}_i \succ \mathrm{D}_i \cup \{j\}$
                ▷ Add $\mathrm{P}_j$ to the set of individuals dominated by $\mathrm{P}_i$.
        **else if** $\mathrm{P}_j \succ \mathrm{P}_i$ **then**
            $\#\mathrm{M}_i \succ \#\mathrm{M}_i + 1$
                ▷ Increase dominators counter for $\mathrm{P}_i$.
**Phase 2: Assign front to the individuals of the** $1_{st}$ **front**
**for** $i \leftarrow 1$ **to** $p$ **do**
    **if** $\#\mathrm{M}_i = 0$ **then** ▷ If individual $\mathrm{P}_i$ belongs to the $1_{st}$ front.
        $\mathrm{T}_i = 1$ ▷ Set individual $i$ to the $1_{st}$ front.

        $\mathrm{F}_1 \leftarrow \mathrm{F}_1 \cup \{i\}$
**Phase 3: Front Assignment**
$t = 1$
**while** $\mathrm{F}_t \neq \{\emptyset\}$ **do**
    $\mathrm{Q} \leftarrow \{\emptyset\}$ ▷ Temporary set used to store individuals of the next front.
    **for each** $i \in \mathrm{F}_t$ **do**
        **for each** $j \in \mathrm{D}_i$ **do**
            $\#\mathrm{M}_i \leftarrow \#\mathrm{M}_i - 1$
            **if** $\#\mathrm{M}_i = 0$ **then**
                $\mathrm{T}_i \leftarrow t + 1$ ▷ Set individual $j$ to the $t+1$ front.
                $\mathrm{Q} \leftarrow \mathrm{Q} \cup \{j\}$
    $t \leftarrow t + 1$ ▷ Increase the front counter.
    $\mathrm{F}_t \leftarrow \mathrm{Q}$
$t_{max} \leftarrow t$
**return** F, T ▷ Returns the Pareto Dominance Ranking and the Front Matrix.

## 11.5 The algorithm for rule base integration with detection of the three issues

To automate the rule base integration and refinement process, we propose an algorithm to detect the issues when an expert rule is integrated into the chronicle rule base. The algorithm checks *Redundancy*, *Conflict*, and *Subsumption* among the chronicle rules and an expert rule. The pseudocode is shown in Algorithm 5. The algorithm runs into six major steps:

- For an expert rule $R_e$, three functions extract different sets of atoms from it. The function *Non-failureEvent* extracts all the non-failure events inside $R_e$. The function *Non-failureTemporalConstraints* extracts the temporal constraints of these non-failure events, and the function *FailureTemporalConstraints* extracts the temporal constraints of the failure.

- For each chronicle rule R in the chronicle rule base $\mathbb{R}$, the three functions that are described in the previous step extract the same types of atoms from it (non-failure events, temporal constraints of the non-failure events, and the temporal constraints of the failure).

- For each chronicle rule R and the expert rule $R_e$, redundancy issue is checked by the algorithm. If they are redundant, the expert rule is removed whilst the redundant chronicle rule is retained in the rule base.

- Rule *Subsumption* is examined after the check of *Redundancy*. For each chronicle rule R and the expert rule $R_e$, if R subsumes $R_e$, then discard $R_e$; If $R_e$ subsumes R, then remove R from the rule base $\mathbb{R}$ and integrate the expert rule $R_e$ into $\mathbb{R}$.

- After the *Redundancy* and *Subsumption* issues are inspected, the last issue to be examined is rule *Conflict*. If the expert rule $R_e$ is in conflict with a chronicle rule R, then remove R and integrate $R_e$ into the rule base. The reason for doing this is based on the assumption that the reliability of an expert rule is always higher than or equal to a chronicle rule when they have *Conflict*. In this context, when there is a rule *Conflict* issue, the expert rule always have the priority of being integrated into the rule base.

- Return the refined rule base $\mathbb{R}'$ as the algorithm output.

By taking the expert rules as input, Algorithm 5 is applied to check whether there exist issues among the expert rules and chronicle rules. As a result, the chronicle rule base is refined and verified progressively for each input expert rule. At last, a refined rule base is obtained, which consists of best-quality chronicle rules as well as expert rules.

**Algorithm 5** The algorithm to detect the three issues when an expert rule is integrated into the chronicle rule base.

**Input:**

$\mathbb{R}$: a chronicle rule base which contains a set of failure chronicles.
$R_e$: an expert rule which is in the form of a failure chronicle.

**Output:**

$\mathbb{R}'$: the integrated rule base.
EE $\leftarrow \emptyset$, TEE $\leftarrow \emptyset$, TFE $\leftarrow \emptyset$, EC $\leftarrow \emptyset$, TEC $\leftarrow \emptyset$, TFC $\leftarrow \emptyset$, $\mathbb{R}' \leftarrow \emptyset$.
EE $\leftarrow$ N$on-failure$E$vent$($R_e$)
▷ Extract all the non-failure events of this expert rule.
TEE $\leftarrow$ N$on-failure$T$emporal$C$onstraints$($R_e$)
▷ Extract the temporal constraints of these non-failure events.
TFE $\leftarrow$ F$ailure$T$emporal$C$onstraints$($R_e$)
▷ Extract the temporal constraints of the failure.
**for each** R $\in \mathbb{R}$ **do**
    EC $\leftarrow$ N$on-failure$E$vent$(R)
    ▷ Extract all the non-failure events of this chronicle rule.
    TEC $\leftarrow$ N$on-failure$T$emporal$C$onstraints$(R)
    ▷ Extract the temporal constraints of the non-failure events in the chronicle rule.
    TFC $\leftarrow$ F$ailure$T$emporal$C$onstraints$(R)
    ▷ Extract the temporal constraints of the failure in the chronicle rule.
    **if** EE = EC, TEE = TEC, TFE = TEC, **then**
        C$hro$R$edundancy$($R_e$, R)  ▷ Rule redundancy issue is detected.
        R$emove$($R_e$, $\mathbb{R}$)  ▷ Remove the redundant expert rule.
    **else if** TFE = TFC, EC = EE, TEC $\subseteq$ TE, **then**
        C$hro$S$ubsumes$($R_e$, R)  ▷ Rule subsumption issue is detected.
        R$emove$(R, $\mathbb{R}$)  ▷ Remove the subsumed chronicle rule from the rule base.
        I$ntegrate$($R_e$, $\mathbb{R}$)  ▷ Integrate the expert rule into the rule base.
    **else if** EE = EC, TE $\subseteq$ TEC, TFE $\neq$ TFC, **then**
        C$hro$C$onflict$($R_e$, R)  ▷ Rule conflict issue is detected.
        R$emove$(R, $\mathbb{R}$)  ▷ Remove the conflict chronicle rule from the rule base.
        I$ntegrate$($R_e$, $\mathbb{R}$)  ▷ Integrate the expert rule into the rule base.
    **else**
        I$ntegrate$($R_e$, $\mathbb{R}$)
    **end if**
**end for each**
$\mathbb{R}' \leftarrow \mathbb{R}$
**return** $\mathbb{R}'$

## 11.6 Experimentation

### 11.6.1 The pruning of chronicle rules

The proposed approach was validated on the UCI SECOM data set. At first, the fitness values of the chronicle rules are computed. Fig. 11.3 shows the fitness values with different *Accuracy* and *Coverage* values. As the fitness of a rule is computed by a product by the two quality measures, its value increases as the values of *Accuracy* and *Coverage* become higher.

With obtaining the fitness values for each chronicle rule, the average fitness value is computed for the set of extracted chronicle rules under specific chronicle support values. After that, the set of rules with the highest average fitness is selected as the input of the fast non-dominated sorting algorithm. The reason to do this is to ensure the rule base obtained from frequent chronicle mining is the most reliable and relevant one. After this step, the fast non-dominated sorting algorithm 4 is applied to the most reliable and relevant chronicle rule base, to discover the best-quality rules out of it. The aim of using this algorithm is to find the rules for which the *Accuracy* and *Coverage* values are larger than others. Thus as output, only the chronicle rules in the first Pareto front are selected.

Table 4 shows the results we get after applying the fast non-dominated sorting algorithm. With different chronicle support thresholds (CST), the table shows the number of extracted rules (NER), the number of rules in the first Pareto front (NRFPF), and the average fitness value for the rules in the first Pareto front (AFFPF).

Table 11.3 – Two rule quality measures under different chronicle support.

| **Support(R)** | **NER** | **NRFPF** | **AFFPF** |
|---|---|---|---|
| 0.5 | 8254 | 18 | 0.7036 |
| 0.6 | 1258 | 8 | 0.7026 |
| 0.7 | 283 | 7 | 0.7220 |
| **0.8** | **30** | **3** | **0.7983** |
| 0.9 | 16 | 3 | 0.7639 |
| 1.0 | 1 | 1 | 0.7895 |

As shown in Table 11.3, the highest AFFPF value is obtained when CST =0.8, and NRFPF = 3. Fig. 11.4 shows the three chronicle rules within the first Pareto front. In the figure, $A_1$, $A_2$, and $A_3$ are non-failure events before the failures, and F indicates a failure event. These rules are in the SWRL rule format that are transformed from the

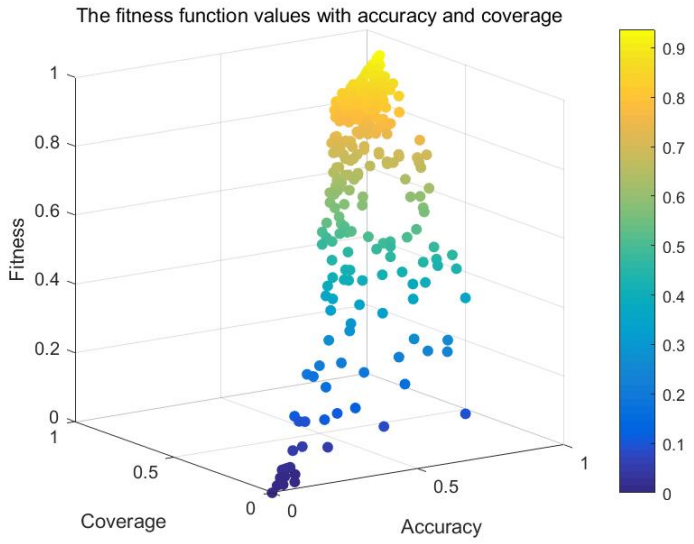



Figure 11.3 – The fitness values of a chronicle rule with different *Accuracy* and *Coverage*.

mined chronicles using Algorithm 1. The nominal attributes are obtained by data discretization, similar to the data pre-processing appraoch introduced in Chapter 9.4.2. In this context, these three rules are used to construct the final chronicle rule base for failure prediction. This allows to make prediction with a set of best-quality chronicle rules.

After rule pruning, the detection process for *Redundancy*, *Conflict*, and *Subsumption* issues is simulated by integrating expert rules into the chronicle rule base.

### 11.6.2 Detection of the three issues for expert rules

Since the frequent chronicle mining algorithm aims to extract closed patterns [SMS+19], *Redundancy*, *Conflict*, and *Subsumption* issues are eliminated for the mined chronicles. These closed patterns are more frequent and informative than the filtered patterns. Therefore, the three rule issues do not apply to the chronicle rule base. In this context, to simulate the expert rules, SWRL rules are created with the aforementioned three issues against the chronicle rules, to evaluate the performance of Algorithm 5.

By applying Algorithm 5, we successfully detect expert rules that have *Redundancy*, *Conflict*, and *Subsumption* issues against the three chronicle rules. Fig. 11.5 shows one expert rule, and the chronicles rules with rule *Redundancy* issue. In this case, both chronicle rule 1 and expert rule 1 have the same sets of events, and the

**Chronicle rule 1:**

*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^ hasP102V(?e1,?v102) ^ swrlb:lessThan(?v102,-0.1187) ^ hasP347V(?e1,?v347) ^ swrlb:greaterThan(?v347,-6.9574) ^ hasP63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^ hasP476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^ hasP204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456) ^ hasP209V(?e1,?v209) ^ swrlb:greaterThan(?v209,23.0750)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

**Chronicle rule 2:**

*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^ hasP102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^ hasP347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574) ^ hasP63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^ hasP129V(?e1,?v129) ^ swrlb:greaterThan(?v129,-0.6605) ^ hasP476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^ hasP204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456) ^ hasP209V(?e1,?v209) ^ swrlb:lessThan(?v209,23.0750)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

**Chronicle rule 3:**

*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^ hasP102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^ hasP347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574) ^ hasP63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^ hasP476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^ hasP32V(?e1,?v32) ^ swrlb:lessThan(?v32,94.3934) ^ hasP204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456) ^ hasP209V(?e1,?v209) ^ swrlb:lessThan(?v209,23.0750)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

Figure 11.4 – The three chronicle rules with best quality, extracted with $chroniclesupport = 0.8$.

*Chronicle Rule 1:*
*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^*
*hasP102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^*
*hasP347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574)^*
*hasP63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^*
*hasP476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^*
*hasP204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456)^*
*hasP209V(?e1,?v209) ^ swrlb:lessThan(?v209,23.0750)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

*Expert Rule 1:*
*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^*
*hasP102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^*
*hasP347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574) ^*
*hasP63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^*
*hasP476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^*
*hasP204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456) ^*
*hasP209V(?e1,?v209) ^ swrlb:lessThan(?v209,23.0750)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

Figure 11.5 – Rule *Redundancy* issue detected between a chronicle rule and an expert rule.

events have the same temporal constraints. Thus a rule *Redundancy* issue is detected.

Fig. 11.6 shows one expert rule and the chronicle rule whom there is a *Conflict* issue with. In the figure, both rules succeed in the same sets of non-failure events and the same temporal constraints for these non-failure events, but with different temporal constraints of a failure. Thus a Conflict issue is detected.

Fig. 11.7 presents one expert rule, and the chronicle rule whom there is a *Subsumption* issue with. In this case, they have the same temporal constraints of a failure, but Chronicle Rule 3 contains additional restrictions on the set of non-failure events. Thus Expert Rule 3 subsumes Chronicle Rule 3.

### 11.6.3 The integration of expert rules with the chronicle rule base

After checking the *Redundancy*, *Conflict*, and *Subsumption* issues, the expert rules are integrated with the mined chronicle rule base. To simulate expert rules, the UCI SECOM data set is split into a training set (80%) and test set (20%). The training set is used to extract chronicle rules and to construct the chronicle rule base and expert

*Chronicle Rule 2:*
*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^*
*hasP102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^*
*hasP347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574) ^*
*hasP63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^*
*hasP129V(?e1,?v129) ^ swrlb:greaterThan(?v129,-0.6605) ^*
*hasP476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^*
*hasP204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456) ^*
*hasP209V(?e1,?v209) ^ swrlb:lessThan(?v209,23.0750)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

---

*Expert Rule 2:*
*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^*
*hasA102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^*
*hasA347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574) ^*
*hasA63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^*
*hasA129V(?e1,?v129) ^ swrlb:greaterThan(?v129,-0.6605) ^*
*hasA476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^*
*hasA204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456) ^*
*hasA209V(?e1,?v209) ^ swrlb:lessThan(?v209,23.0750)*
***-> hasMinF(?e1, 3) ^ hasMaxF(?e1, 2663881)***

Figure 11.6 – Rule *Conflict* issue detected between a chronicle rule and an expert rule.

*Chronicle Rule 3:*
*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^*
*hasP102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^*
*hasP347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574) ^*
*hasP63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757) ^*
*hasP476V(?e1,?v476) ^ swrlb:lessThan(?v476,137.4436) ^*
*hasP32V(?e1,?v32) ^ swrlb:lessThan(?v32,94.3934) ^*
*hasP204V(?e1,?v204) ^ swrlb:lessThan(?v204,4999.2456) ^*
*hasP209V(?e1,?v209) ^ swrlb:lessThan(?v209,23.0750)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

---

*Expert Rule 3:*
*ManufacturingProcess(?s) ^ hasEvent(?s,?e1) ^*
*hasA102V(?e1,?v102) ^ swrlb:greaterThan(?v102,-0.1187) ^*
*hasA347V(?e1,?v347) ^ swrlb:lessThan(?v347,-6.9574) ^*
*hasA63V(?e1,?v63) ^ swrlb:greaterThan(?v63,94.8757)*
***-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2663881)***

Figure 11.7 – Rule *Subsumption* issue detected between a chronicle rule and an expert rule.

rule base. After that, we select a set of chronicle rules that have the highest fitness values. Since these high-fitness chronicle rules are likely to have a higher chance to validate the data sequences in the test set, they are considered as expert rules.

To simulate the rule base integration process, the frequent chronicle mining algorithm is applied to the training set. Then the fast non-dominated sorting algorithm is used to obtain a set of best-quality chronicle rules. We then select the chronicle rules within the first Pareto front. Since the selected rules Pareto-dominate all the rest ones, we treat them as expert rules. All the rules in the other Pareto fronts construct a chronicle rule base.

After that, the expert rules are integrated into the chronicle rule base. During the integration process, Algorithm 5 is used to detect *Redundancy*, *Conflict*, and *Subsumption*issues. If no aforementioned issues are detected, the expert rule is integrated into the chronicle rule base. After that, the average fitness of the updated rule base is computed. With each integration, the updated rule base is launched for failure prediction against the test set. Fig. 11.8 shows the obtained average fitness values with different number of expert rules as input. For the ease of visualization, we choose $chroniclesupport = 0.6$ and show the change of fitness values. It can be observed that as more expert rules are integrated into the chronicle rule base,

the average fitness increases. This is because as more expert rules are integrated into the rule base, there are more chances for the updated rule base to validate the data sequences in the test set. As a result, as more expert rules are launched for reasoning, the integrated rule base has more chances to predict the failures correctly. Thus the overall fitness of the integrated rule base increases.

However, there is one exception where the average fitness of the updated rule base does not increase (number of expert rules = 4). It is because the fitness value of the new expert rule for the test set is lower than the average value of the whole rule base. Thus the fitness value of the updated rule base is decreased. When the antecedent of the input expert rule only covers a few data sequence in the test set, or most of the temporal constraints of the failures are out of the range that is predicted in the expert rule, the input expert rule shows a low fitness value.

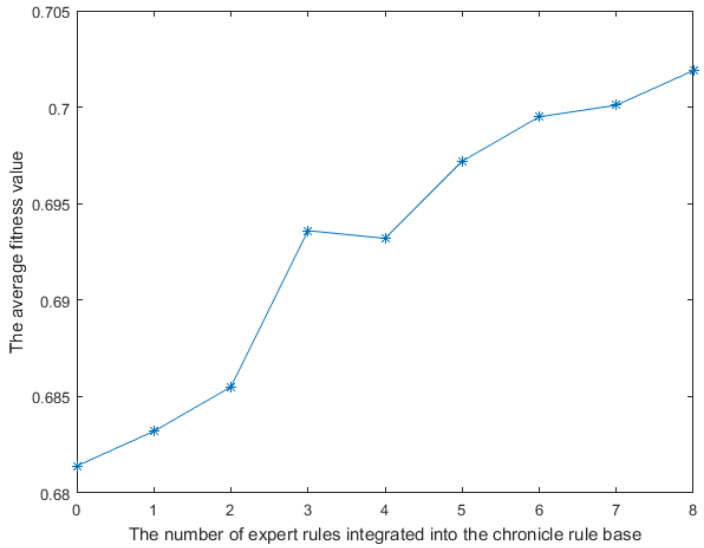



Figure 11.8 – The average fitness values of the rules in the integrated rule base with different number of expert rules.

In this way, Algorithm 5 checks issues that may occur when an expert rule is proposed by domain experts. After each check, the proposed algorithm ensures the rule base is updated progressively according to different conditions. This mechanism ensures the rule base is verified and validated with each time of update.

## 11.7 Summary

In this chapter, a novel rule base refinement approach for knowledge-based predictive maintenance systems is proposed. The proposed approach uses rule quality measures as criteria to refine rule bases that consist of rules collected from heterogeneous sources. Within the approach, a multi-objective optimization method for the pruning and reduction of rule bases is used, with *Accuracy* and *Coverage* as the two considered criteria for it. A fast non-dominated sorting algorithm is used to obtain the ranking of Pareto dominance for chronicle rules. After that, the rules within the first front are selected to perform rule-based reasoning. In this way, a set of chronicle rules that have the best quality are launched for failure prediction.

After rule pruning, a rule integration method for combining chronicle rules and expert rules is introduced. The integration method detects rule *Redundancy*, *Conflict*, and *Subsumption* issues among expert rules and chronicle rules. After that, the expert rules that do not have these three issues are integrated with the chronicle rules. In this way, the rule bases are progressively updated to achieve better performance for failure prediction. The proposed approach is validated on a real-world industrial data set.

# Chapter 12

# The software prototype: KSPMI

## Contents

From chapter 8 to chapter 11, we have introduced different steps within a predictive maintenance process. In chapter 8, an ontology-based framework is developed as the foundation of a knowledge-based predictive maintenance system, which contains a domain-level ontology named MFPO. In chapter 9, a novel hybrid semantic approach is proposed to automate machinery failure prediction tasks, which is based on the combined use of chronicles and semantic technologies. In chapter 10, the severity levels of failures are assessed by two clustering tools: fuzzy *c*-means (FCM) clustering [BEF84] and evidential *c*-means (ECM) clustering. In chapter 11, a novel rule base refinement approach is proposed to update the rule base with expert rules as input.

To realize and automate the aforementioned predictive maintenance pipeline, we have developed a software prototype named Knowledge-based System for Predictive Maintenance in Industry 4.0 (KSPMI)[1] [2]. The software uses both inductive approaches (chronicle mining and machine learning) and deductive approaches (domain ontologies and ontology reasoning) to analyze industrial data and to predict future failures. In this chapter, we first introduce the development environment and tools for the software prototype. Then we introduce the core functionalities of it by following the key steps within the predictive maintenance pipeline: chronicle mining, SWRL rule transformation/generation, SWRL rule pruning, experience capitalization/expert rules integration, and failure prediction.

## 12.1 Software development environment and tools

During the development of KSPMI, several software and tools are used. They are listed as follows:

- Java Platform, Standard Edition (Java SE) [3]. Java SE is a computing platform for developing and deploying portable codes for desktop and server applications. Java is an object-oriented programming language designed for building up mobile applications, data processing tools, and embedded systems.

- Eclipse[4]. It is an integrated development environment (IDE) for computer

[1] The souce code of KSPMI can be found at: https://github.com/caoppg/KSPMI.git

[2] A demonstration video for KSPMI can be found at: https://sites.google.com/view/qiushi-phd-thesis/home

[3] https://www.oracle.com/java/technologies/javase-downloads.html

[4] https://www.eclipse.org/

programming. With its primary use for developing Java applications, Eclipse consists of a workspace and an extensible plug-in system for users to customize their development environment.

- Protégé[5]. Developed by Stanford University, Protégé is a free and open-source software for developing knowledge-based systems and editing ontologies. It can be extended by a set of plug-in applets and can be transferred into a Java-based Application Programming Interface (API) for the development of Java-based intelligent systems. During our development phase, three Protégé-based APIs have been used: OWL API[6], SWRL API[7], and SQWRL API[8].

- Drools[9]. It is a business rule engine that provides forward and backward chaining inference capabilities. Drools have been widely used for developing production rule systems. In KSPMI, Drools is used to execute SWRL rules for ontology reasoning, which aims to predict the time of machinery failures.

- SPMF[10]. SPMF is an open-source data mining library written in Java, and it provides a set of pattern mining algorithms to discover patterns out of data. The main advantage of using SPMF is the source code of each algorithm in it can be easily integrated with other programs that are written in Java.

In the next sections, we introduce the key functionalities and main Graphical User Interfaces (GUI) of KSPMI.

## 12.2 The chronicle mining GUI

The running of KSPMI starts with the chronicle mining phase on input data. Within this phase, the frequent chronicle mining algorithm introduced in [SMS+19] is used to extract failure chronicles out of data. These failure chronicles describe a set of ordered events and the temporal constraints of failures. Fig. 12.1 shows the chronicle mining GUI of KSPMI.

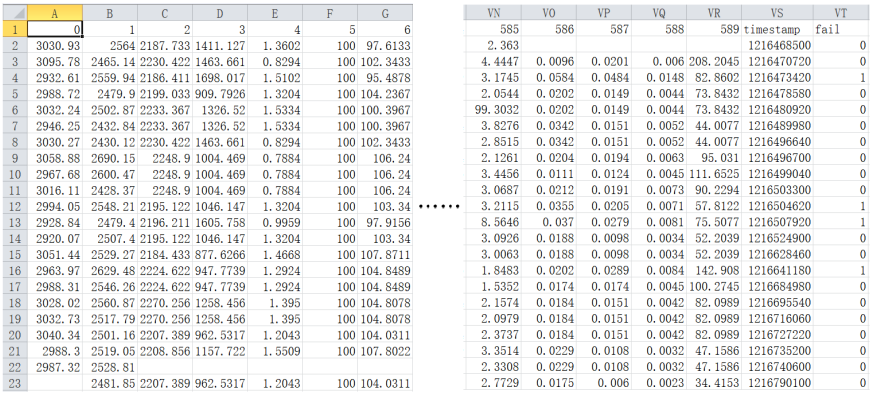
[5]https://protege.stanford.edu/
[6]http://owlapi.sourceforge.net/
[7]https://github.com/protegeproject/swrlapi
[8]https://github.com/protegeproject/swrlapi/wiki/SQWRL
[9]https://www.drools.org/
[10]https://www.philippe-fournier-viger.com/spmf/


Figure 12.1 – The chronicle mining GUI of KSPMI.

The software takes sequential data sets as input. In the input data set, it is required that events are associated with timestamps, and each test point is given a binary label (e.g., –1 corresponds to a normal status and 1 corresponds to a failed status). Fig. 12.2 shows an excerpt from an input data set. The rows in the data set stand for test points. Every test point consists of 589 attributes, where each attribute is associated with its numerical value. The values in the second last column are the timestamps when these test points are recorded. Since the timestamps are in the ascending order, the test points are considered as a sequence of events that are timely ordered. A binary label is given in the last column for each row, which indicates a pass (0) or fail (1) of the recorded test point.

In the chronicle mining interface, a set of extracted failure chronicles are displayed in the table within Figure 12.1. There are three attributes associated with each failure chronicle: support, accuracy, and coverage. Among them, the support value is calculated based on Definition 6. Accuracy and coverage are computed by equation 10.14 and 10.15. With clicking on each table entry, the values of these two rule quality measures are shown on the right side. At the same time, the graph structure of the selected failure chronicle is also displayed. Within each graph, the event with integer 0 indicates a failure. The integers in other events are the nominal attributes we obtain after data discretization.

At the bottom of the chronicle mining interface, users can press the button to



| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 2 | 3030.93 | 2564 | 2187.733 | 1411.127 | 1.3602 | 100 | 97.6133 |
| 3 | 3095.78 | 2465.14 | 2230.422 | 1463.661 | 0.8294 | 100 | 102.3433 |
| 4 | 2932.61 | 2559.94 | 2186.411 | 1698.017 | 1.5102 | 100 | 95.4878 |
| 5 | 2988.72 | 2479.9 | 2199.033 | 909.7926 | 1.3204 | 100 | 104.2367 |
| 6 | 3032.24 | 2502.87 | 2233.367 | 1326.52 | 1.5334 | 100 | 100.3967 |
| 7 | 2946.25 | 2432.84 | 2233.367 | 1326.52 | 1.5334 | 100 | 100.3967 |
| 8 | 3030.27 | 2430.12 | 2230.422 | 1463.661 | 0.8294 | 100 | 102.3433 |
| 9 | 3058.88 | 2690.15 | 2248.9 | 1004.469 | 0.7884 | 100 | 106.24 |
| 10 | 2967.68 | 2600.47 | 2248.9 | 1004.469 | 0.7884 | 100 | 106.24 |
| 11 | 3016.11 | 2428.37 | 2248.9 | 1004.469 | 0.7884 | 100 | 106.24 |
| 12 | 2994.05 | 2548.21 | 2195.122 | 1046.147 | 1.3204 | 100 | 103.34 |
| 13 | 2928.84 | 2479.4 | 2196.211 | 1605.758 | 0.9959 | 100 | 97.9156 |
| 14 | 2920.07 | 2507.4 | 2195.122 | 1046.147 | 1.3204 | 100 | 103.34 |
| 15 | 3051.44 | 2529.27 | 2184.433 | 877.6266 | 1.4668 | 100 | 107.8711 |
| 16 | 2963.97 | 2629.48 | 2224.622 | 947.7739 | 1.2924 | 100 | 104.8489 |
| 17 | 2988.31 | 2546.26 | 2224.622 | 947.7739 | 1.2924 | 100 | 104.8489 |
| 18 | 3028.02 | 2560.87 | 2270.256 | 1258.456 | 1.395 | 100 | 104.8078 |
| 19 | 3032.73 | 2517.79 | 2270.256 | 1258.456 | 1.395 | 100 | 104.8078 |
| 20 | 3040.34 | 2501.16 | 2207.389 | 962.5317 | 1.2043 | 100 | 104.0311 |
| 21 | 2988.3 | 2519.05 | 2208.856 | 1157.722 | 1.5509 | 100 | 107.8022 |
| 22 | 2987.32 | 2528.81 | | | | | |
| 23 | | 2481.85 | 2207.389 | 962.5317 | 1.2043 | 100 | 104.0311 |

| VN | VO | VP | VQ | VR | VS | VT |
|---|---|---|---|---|---|---|
| 585 | 586 | 587 | 588 | 589 | timestamp | fail |
| 2.363 | | | | | 1216468500 | 0 |
| 4.4447 | 0.0096 | 0.0201 | 0.006 | 208.2045 | 1216470720 | 0 |
| 3.1745 | 0.0584 | 0.0484 | 0.0148 | 82.8602 | 1216473420 | 1 |
| 2.0544 | 0.0202 | 0.0149 | 0.0044 | 73.8432 | 1216478580 | 0 |
| 99.3032 | 0.0202 | 0.0149 | 0.0044 | 73.8432 | 1216480920 | 0 |
| 3.8276 | 0.0342 | 0.0151 | 0.0052 | 44.0077 | 1216489980 | 0 |
| 2.8515 | 0.0342 | 0.0151 | 0.0052 | 44.0077 | 1216496640 | 0 |
| 2.1261 | 0.0204 | 0.0194 | 0.0063 | 95.031 | 1216496700 | 0 |
| 3.4456 | 0.0111 | 0.0124 | 0.0045 | 111.6525 | 1216499040 | 0 |
| 3.0687 | 0.0212 | 0.0191 | 0.0073 | 90.2294 | 1216503300 | 0 |
| 3.2115 | 0.0355 | 0.0205 | 0.0071 | 57.8122 | 1216504620 | 1 |
| 8.5646 | 0.037 | 0.0279 | 0.0081 | 75.5077 | 1216507920 | 1 |
| 3.0926 | 0.0188 | 0.0098 | 0.0034 | 52.2039 | 1216524900 | 0 |
| 3.0063 | 0.0188 | 0.0098 | 0.0034 | 52.2039 | 1216628460 | 0 |
| 1.8483 | 0.0202 | 0.0289 | 0.0084 | 142.908 | 1216641180 | 1 |
| 1.5352 | 0.0174 | 0.0174 | 0.0045 | 100.2745 | 1216684980 | 0 |
| 2.1574 | 0.0184 | 0.0151 | 0.0042 | 82.0989 | 1216695540 | 0 |
| 2.0979 | 0.0184 | 0.0151 | 0.0042 | 82.0989 | 1216716060 | 0 |
| 2.3737 | 0.0184 | 0.0151 | 0.0042 | 82.0989 | 1216727220 | 0 |
| 3.3514 | 0.0229 | 0.0108 | 0.0032 | 47.1586 | 1216735200 | 0 |
| 2.3308 | 0.0229 | 0.0108 | 0.0032 | 47.1586 | 1216740600 | 0 |
| 2.7729 | 0.0175 | 0.006 | 0.0023 | 34.4153 | 1216790100 | 0 |

Figure 12.2 – An excerpt of an input data set.

trigger the SWRL rule transformation phase. After clicking this button, Algorithm 1, which is introduced in Chapter 9, is used to generate a set of predictive SWRL rules from the extracted failure chronicles. These SWRL rules construct a chronicle rule base.

## 12.3 The SWRL rules transformation and rule pruning GUI

After the chronicle mining step, the extracted failure chronicles are transformed into a set of SWRL rules. The SWRL rules transformation and rule pruning GUI allows users to visualize the transformed rules and to select the best-quality rules by using a multi-objective optimization approach. Fig. 12.3 shows this GUI.

Within this interface, users can first display the transformed rule base by clicking “Display”. Then, the syntax of rules is presented in the text area on the left side. After that, the fitness function of the multi-objective optimization approach can be shown by clicking the “Fitness Function” button. The fitness function we use in this phase is the product of rule accuracy and coverage, as introduced by Equation 11.6.

The button “Rule pruning” launches the pruning of the rule base by using the fast non-dominated sorting algorithm on rule quality measures. After the pruning phase, a subset of best-quality rules are displayed on the left side, and the objective function values of them can be plotted by clicking the button “Plot the Best Rules”. Fig. 12.4 shows the rule pruning results obtained after clicking the “Plot the Best

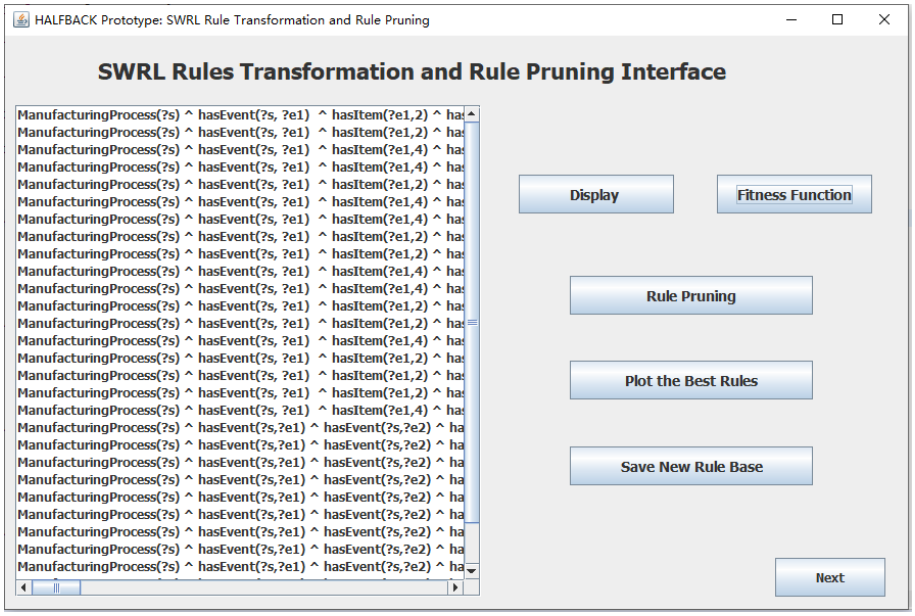



Figure 12.3 – The SWRL rules transformation and rule pruning GUI.

Rules" button.

Up to this step, we have completed rule transformation and pruning.

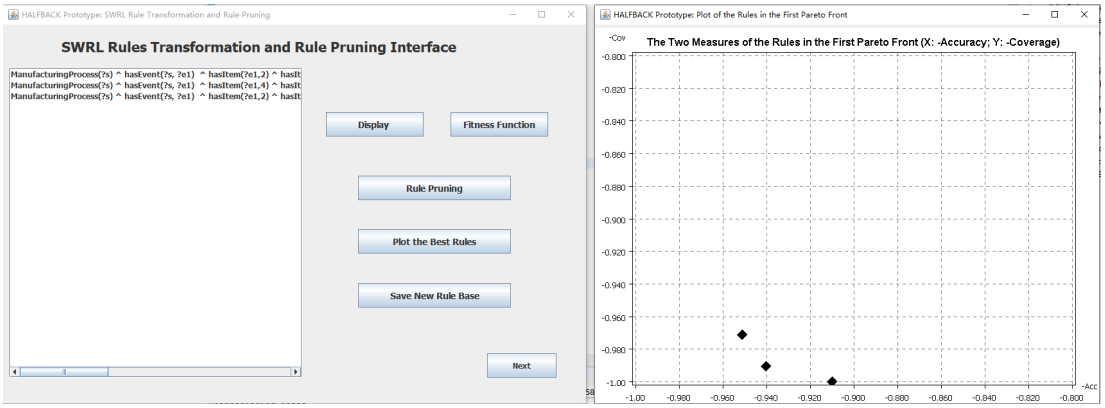



Figure 12.4 – The rule pruning results.

## 12.4 The experience capitalization GUI

After obtaining a set of best-quality rules, we can proceed to the experience capitalization phase, where expert rules are taken as input to the program. The goal of integrating expert rules into the system is to improve the overall fitness of the whole rule base. To enable this step, an experience capitalization GUI has been developed to allow users to interact with the expert rule base. Also, the system detects possible issues that may occur during this integration process, such as rule redundancy, conflict, and subsumption.

Fig. 12.5 shows the experience capitalization GUI of KSPMI. Users can first open the expert rule file, which is stored separately from the chronicle rules. Fig. 12.6 shows an example of an expert rule file. The expert rules are the same format as the chronicle rules, with differences of rule atoms within either antecedent or consequent part.

The expert rules can be integrated into the chronicle rule base by clicking the "Input Expert Rule" button. After clicking this button, the system tries to push the input expert rule into the chronicle rule base. If there is an issue detected, corresponding actions are automatically performed according to the decision making process in Algorithm 5.

On the other hand, if no issue detected, the input expert rule is directly integrated into the chronicle rule base. At last, the integrated rule base is updated. Fig. 12.5 shows a situation where a conflict issue is detected. The system automatically removes the conflict chronicle rule, and add the expert rule into the rule base. The reason for is action is based on the assumption that we assign the fitness value of all expert rules to 1, which means the expert rules are always more reliable than chronicle rules.

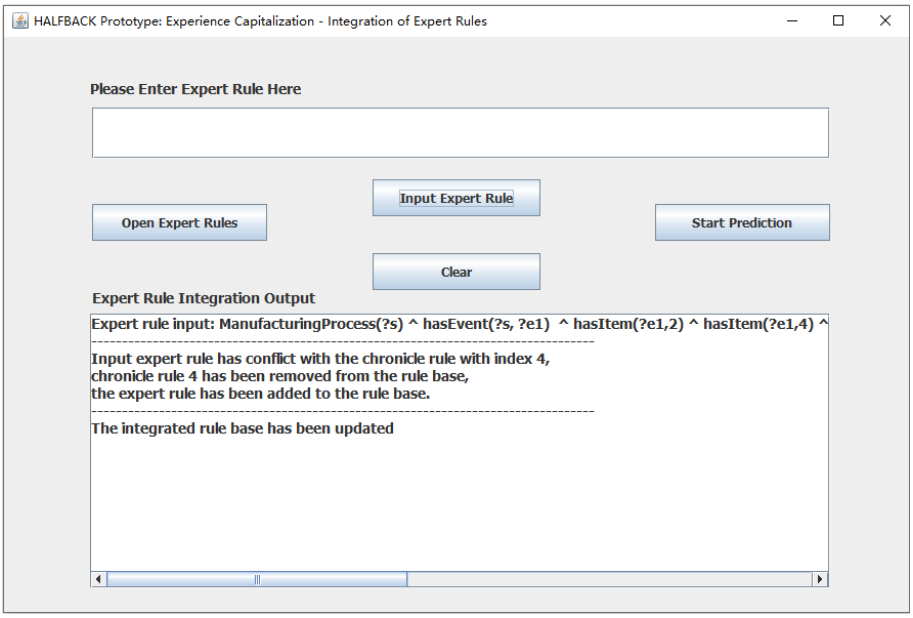



Figure 12.5 – The experience capitalization GUI.

```
Expert rule 1:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,11) ^ hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2656921)

Expert rule 2:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,11) ^ hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2656921)

Expert rule 3:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^ hasItem(?e1,11) ^
hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 3) ^ hasMaxF(?e1, 1567841)

Expert rule 4:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7)
-> hasMinF(?e1, 3) ^ hasMaxF(?e1, 1567841)

Expert rule 5:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,11) ^ hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2656921)

Expert rule 6:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,8) ^ hasItem(?e1,9) -> hasMinF(?e1, 115) ^
hasMaxF(?e1, 784613)

Expert rule 7:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,8) -> hasMinF(?e1, 525) ^ hasMaxF(?e1, 254651)

Expert rule 8:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,6) ^ hasItem(?e1,7) ^
hasItem(?e1,8) -> hasMinF(?e1, 310) ^ hasMaxF(?e1, 421974)
```

Figure 12.6 – An example expert rule base.

## 12.5 The ontology reasoning and failure prediction GUI

After the experience capitalization process, we can proceed to the failure prediction step. The failure prediction work is achieved by using the Drools rule engine to perform ontology reasoning on the data that is populated in the MFPO ontology (introduced in Chapter 8). After prediction, a SQWRL query is created to retrieve the prediction results. The SQWRL language takes an antecedent of a SWRL rule and effectively treats it as a pattern specification for a query. To extract the results, a SQWRL query replaces a rule consequent with a retrieval specification.

Fig. 12.7 shows the first round of prediction results. In this round, five expert rules are taken as input to the experience capitalization process. After the rule issue detection step, the updated rule base used for prediction consists of 6 rules. As a result, 13 rows of prediction have been obtained after ontology reasoning. They are shown in the table at the bottom part of the GUI.

After the first round of prediction, we continue integrating more expert rules into the rule base. In the second round, the updated rule base consists of 9 rules (2 chronicle rules and 7 expert rules), as shown in Fig. 12.8. Then we use this new rule base to perform the second round of prediction. As a result, we obtain 39 rows of results indicating 39 failures. This proves that as more expert rules are integrated into the rule base, the coverage of the whole rule base improves. In this way, the rule base is progressively updated with expert rules as input, and more potential failures are detected.

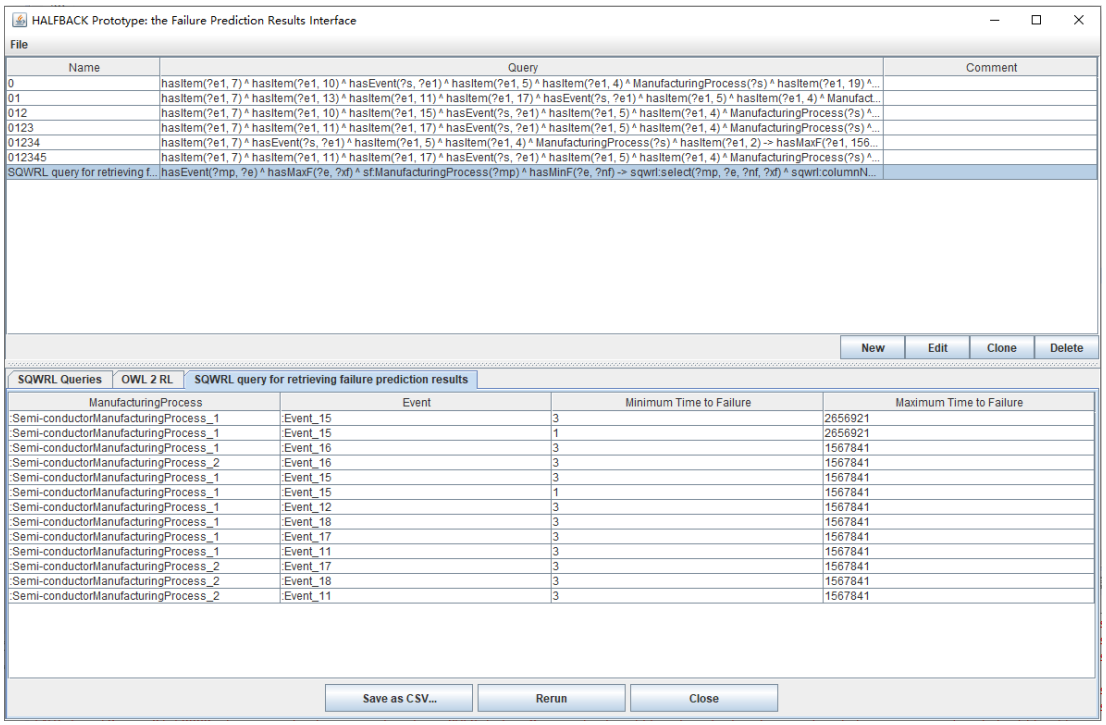



Figure 12.7 – The failure prediction results with 6 rules.

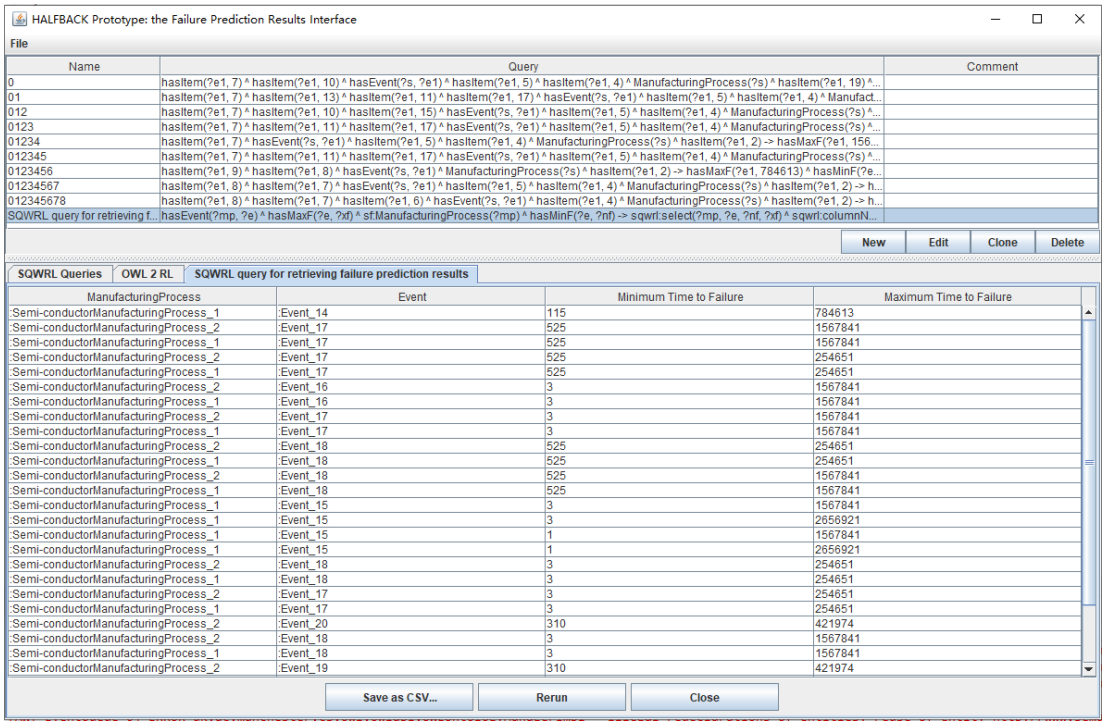



Figure 12.8 – The failure prediction results with 9 rules.

## 12.6 Summary

This chapter introduces a software prototype we have developed for the HALFBACK project. The software enables the whole predictive maintenance process by starting with chronicle mining on industrial data sets, after which a set of failure chronicles are extracted. Then, the novel semantic approach introduced in Chapter 9 is used to transform the chronicles into predictive SWRL rules. The SWRL rules reason on the data that is populated in the domain ontologies developed in Chapter 8. At last, the rule base pruning and integration approach in Chapter 11 are used to update the rule base automatically. Simulated expert rules are taken as input to the rule base integration process. At last, the integrated rule base is used together with domain ontologies to perform ontology reasoning, for the goal of failure prediction.

# Chapter 13

# Conclusions and future work

## Contents

In the manufacturing domain, anomalies such as machinery faults and failures may cause a company high economic costs. To ensure high productivity, availability, and efficiency of manufacturing processes, the detection of harmful tendencies and conditions of production lines is a crucial issue for manufacturers. This appeals to the implementation of *predictive maintenance* tasks, which aims to identify the operating conditions of machinery in production lines. Predictive maintenance is crucial for improving the productivity and availability of production systems. According to the current state of machinery, if any fault or failure exists, a diagnosis task can be launched to determine the causes of it. Also, based on the characteristics of the fault or failure, analysis of how they will propagate and evolve over time can be performed. The use of predictive maintenance techniques has several advantages, such as improved machine availability, improved production efficiency, and reduced maintenance cost.

With the vision of Industry 4.0, the manufacturing industry today is benefiting from a trend of automation in data exchange. Cyber-Physical Systems (CPS) are central to this vision and are entitled to be part of smart factories, and production facilities are able to exchange information with autonomy and intelligence. In smart factories, manufacturing machines are connected building CPSs, which are a new class of engineered systems that offer close interaction between cyber and physical components. The automatic exchange and analysis of data open up opportunities for manufacturers to further optimize the production processes. Collecting data from various components of a production line and analyzing them in a scalable Cloud infrastructure can significantly improve the productivity, reliability, and availability of production systems in heterogeneous environments. However, the utilization of these advanced technologies not only offers the aforementioned benefits to manufactures but also brings them challenges, such as the management of big and heterogeneous data generated by networked machines and sensors.

This vision has been realized within the European Interreg HALFBACK project [1], which aims to to assure highly available manufacturing processes, by forecasting failures of machines, tools, product quality loss, resource flow problems, etc. and by scheduling maintenance, component replacing, process re-planning, and even take over the production by another factory, in an optimized and intelligent way.

Within the HAFLBACK Project, the management of heterogeneous industrial data is considered as a challenging task in the context of predictive maintenance.

[1] http://halfback.in.hs-furtwangen.de/home/

Normally, when a deterioration trend of a machine is detected, highly experienced machine operators are capable of performing appropriate actions to prevent the outage situation of the production system. However, as the structure and behavior of production systems are getting more and more complex, the volume of machine operating data grows significantly. Thus it is possible that the domain professionals fail to respond to a machinery fault or failure timely and accurately. For this reason, manufacturing companies are searching for solutions through which they can manage this heterogeneous industrial data efficiently and perform prognostic and diagnostic tasks intelligently.

On the other hand, as CPSs become more and more complex, the knowledge required for system operation and maintenance increases in complexity. In this context, standard and well-defined models for capturing this complex knowledge is in high demand. To develop such a model, domain knowledge of manufacturing and predictive maintenance is required to be structured in a formal way, thus making this knowledge usable by a CPS. Furthermore, as the manufacturing domain is becoming more and more knowledge-intensive, uniform knowledge representation of physical resources and reasoning capabilities are needed to automate the decision-making processes in CPSs. These decision-making processes include automatic resource integration, anomaly prediction and diagnosis, maintenance scheduling, and process re-planning. To realize this vision, semantic technologies have shown promising results when formalizing knowledge about predictive maintenance tasks in various domains.

To address the aforementioned challenges, in this thesis, a novel semantic approach has been proposed to automate and facilitate predictive maintenance tasks in manufacturing processes. It enables the representation of data mining results in a formal and structured format, thus facilitating the understanding and exploitation of the extracted knowledge. In this way, data results are capable of being interpreted by both users and machines for it to further enrich and improve the knowledge bases in knowledge-based predictive maintenance systems.

## 13.1 Contributions of this thesis work

In this subsection, the contributions of this thesis are recalled. For each part of the contribution, the achieved results and general conclusions are presented.

In Chapter 8, an ontological framework is presented. The ontological frame-

work is the core component of a knowledge-based predictive maintenance system. It is developed based on an ontological representation of predictive maintenance knowledge in the manufacturing domain. Inside the framework, a set of ontologies with different levels of specialization has been developed. It includes a core reference ontology for representing general predictive maintenance concepts and relations and a set of domain ontologies for formalizing domain-specific knowledge of manufacturing and condition monitoring. A case study on a conditional maintenance task of bearings in rotating machinery is presented. In this case study, the ontological framework is jointly used with Semantic Web Rule Language (SWRL) rules to perform rule-based reasoning. The rule-based reasoning results enables the identification of the conditions of these bearings in rotating machinery.

In Chapter 9, a novel hybrid semantic approach for machinery failure prediction is proposed. Within the approach, chronicle mining is used to predict the future failures of the monitored industrial machinery, and domain ontologies (introduced in Chapter 8 with their rule-based extensions are used to predict temporal constraints of failures and to represent the predictive results formally. A case study on a real-world data set is used to demonstrate our approach in detail. The evaluation of results shows that the developed domain ontologies are free of bad practices in structural, functional, and usability-profiling dimensions. The constructed SWRL rules posses more than 80% of *True Positive Rate*, *Precision*, and *F-measure*, which show promising performance in failure prediction.

Chapter 10 is devoted to the assessment of failure criticality. The assessment of failures criticality levels enables the launching of warning signals with different levels. By then, machine operators can prioritize maintenance actions for higher-criticality-level failures compared to lower-level ones. In this thesis, a novel approach to classify failures according to their criticality levels is proposed. The approach is based on clustering and semantic technologies, within which fuzzy and evidential clustering techniques are used to learn the criticality of the failures based on machine historical data, and semantic technologies use the machine learning results to predict the time of failures and the criticality of them.

Chapter 11 introduces a novel rule base refinement approach for integrating rule bases that are obtained from heterogeneous sources. It consists of a rule pruning/reduction method that is applied to the rule base obtained from frequent chronicle mining, and a rule base refinement method for integrating chronicle rules with expert rules. From the work of Chapter 9, a set of chronicle rules are mined from

industrial data sets. To reduce the number of extracted rules and to achieve the best quality of the rule base, a multi-objective optimization approach is applied. The approach aims to maximize rule *Accuracy* and *Coverage* for obtaining a set of rules with the best quality. Within the approach, a fast non-dominated sorting procedure is used to obtain the ranking of Pareto dominance for chronicle rules. After that, the rules within the first front are selected to perform rule-based reasoning. In this way, a set of chronicle rules that have the best quality are launched for failure prediction. On the other hand, to enhance the performance of failure prediction, experts' experience is capitalized in the form of expert rules when the selected chronicle rules fail to provide correct decisions. When these expert rules are integrated with the chronicle rules, they may suffer from several issues such as *Redundancy*, *Contradiction*, and *Subsumption*. In this context, the rule integration method aim to detect the three issues and update the rule base. Experimental results show that the proposed approach ensures a rule base to be progressively updated, and a better failure prediction performance is achieved, compared to the rule base obtained merely from chronicle mining. The proposed method is validated on a real-world data set, which is collected from a semi-conductor manufacturing process.

Chapter 12 presents KSPMI, which is a software prototype using both inductive approaches (chronicle mining and machine learning) and deductive approaches (domain ontologies and ontology reasoning) to analyze industrial data and to predict future failures. KSPMI is an expert system developed in Java, and it uses sequential data sets as input to extract frequent patterns and predictive rules for failure prediction. The prototype aims to implement all the proposed approaches on industrial data sets, for automating the predictive maintenance of manufacturing processes. The software enables the whole predictive maintenance process by starting with chronicle mining on industrial data sets, after which a set of frequent chronicles are extracted. Then the novel semantic approach introduced in Chapter 9 is used to transform the chronicles into predictive SWRL rules. The SWRL rules reason on the data that is populated in the developed ontologies (Chapter 8). At last, the rule base pruning and integration approach in Chapter 11 are used to refine and update the rule base automatically. Simulated expert rules are taken as input for the rule base integration process. At last, the integrated rule base is used together with domain ontologies to perform ontology reasoning, for the goal of failure prediction.

## 13.2 Perspectives

The contributions summarized in the previous section may induce potential future research. In this section, we detail the following perspectives:

- The first future work is the evolution of the ontology and the rule base. Since the manufacturing domain is highly-dynamic, a predictive maintenance system should be able to adapt itself to dynamic situations over time, for example, the change of context. This requires the ontology and the rule base to be capable of coping with the dynamic change of knowledge. To deal with this issue, knowledge base evolution solutions are going to be proposed: the ontology developed in Chapter 8 should be able to adapt itself efficiently to the changes with using ontology evolution techniques [SMMS02], and the rule base extracted from chronicle mining (introduced in Chapter 9) should be updated according to the change of context, by implementing contextual reasoning [BBH+10]. By this, a knowledge-based predictive maintenance system is able to be context-aware, by adapting itself to different physical context, computational context, and user context. For example, the chronicles rule bases extracted from a same machine may differ from each other due to the different humidity and temperature levels of a manufacturing cell. By implementing contextual reasoning, different sets of rules can be selected for ontology reasoning, according to the specific physical context of the operating machine. This ensures a more precise and accurate predictive maintenance strategy.

- The second future work is the consideration of more rule quality measures. In Chapter 10, a multi-objective optimization approach which aims to select rules with maximum *Accuracy* and *Coverage* has been proposed. In the future, we aim to involve more rule quality measures for rule pruning, such as *Association*, *Information*, and *Logical Sufficiency* [AC01]. Among them, *Association* is a measure for indicating a relationship between the classification for the columns and the rows in the contingency table. By computing a rule *Association*, a $G^2$ *likelihood ratio* measure can be obtained. This measure evaluates the distance between two distributions: the observed frequency distribution of examples among classes satisfying the rule R and the expected frequency distribution of the same number of examples under the assumption that the

rule R selects examples randomly [AC01]. It helps to derive the levels of association between the antecedent and consequent of a rule R. *Information* measures the amount of entities a R needs to encode for correctly classifying an instance into a certain class. Statistical values such as *Information Score* can be computed for a rule R, to evaluate how much information it can contribute for a classification problem [KB91]. It is computed as $Q_{IS} = -log\frac{N_a}{N} + log\frac{N_{af}}{N_a}$, where the values of $N_a$, $N_{af}$, and N can be computed from the contingency table 10.3. *Logical Sufficiency* is a standard likelihood ratio statistics that aims to measure whether a rule R is suitable to be used for classifying a certain class. If the *Logical Sufficiency* value of a rule R is large with regard to class C, then this rule is considered encouraging for classifying the training examples to class C. *Logical Sufficiency* is computed as $Q_{LS} = \frac{n_{af}/n_f}{n_{a\bar{f}}/n_{\bar{f}}}$.

- The third future perspective is the evaluation of rule quality for the expert rules. In Chapter 11, a rule base refinement approach is proposed for integrating rules that are obtained from heterogeneous sources. However, the proposed approach is based on the hypothesis that the expert rules are more reliable than the chronicle rules. Indeed, due to the dynamic and uncertain characteristics of the manufacturing domain, experts may provide erroneous expert rules for failure prediction. The erroneous expert rules may lead to incorrect failure prediction results. To address this issue, an evaluation process is needed to examine the performance of the expert rules. In this way, a set of best-quality rules is selected not from the chronicle rule base, but from the integrated rule base, which ensures higher performance in prediction.

- The fourth perspective is the capability of the system to handle real-time data. Since the manufacturing domain is highly-dynamic, how to process real-time and heterogeneous data streams is a crucial concern to manufactures. However, the proposed approach uses the classical ontology reasoning techniques, which can not deal with highly dynamic data in a timely fashion. To cope with this issue, stream reasoning techniques should be adopted to reason upon a variety of highly dynamic data [DDVvHB17]. In stream reasoning, rich query languages are provided by stream reasoners to continuously query data streams. In this way, predictive maintenance systems will be able to detect and predict machinery failures in real-time.

# Part IV

# Appendix

# Appendix A

# The fuzzy *c*-means clustering algorithm

This Appendix gives the theory of fuzzy $c$-means (FCM) clustering algorithm. The notions and concepts presented in this Appendix are taken from the research papers [BEF84] and [PB95].

Similar to $k$-means algorithms [ARS97], the FCM algorithm aims to minimize the objective function defined as

$$\mathrm{J} = \sum_{j=1}^{\mathrm{N}} \sum_{i=1}^{\mathrm{C}} u_{ij}^{m} || \, x_j - c_i \, ||^2, \tag{A.1}$$

where

- $u_{ij}$ represents the degree of membership of the data point $x_j$ in the $i$th cluster.
- $c_j$ stands for the d-dimension center of the cluster.
- $m$ is the fuzzifier which is any real number greater than 1.
- $||\cdot||$ denotes any norm expressing the similarity between any measured data and the center.

The variable $u_{ij}$ is defined as the following equation:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} (\frac{||x_i - c_j||}{||x_i - c_k||})^{\frac{2}{(m-1)}}}. \tag{A.2}$$

$u_{ij}$ indicates the *degree of membership* of a data point to certain clusters. It is assigned inversely to the distance from data point $x_j$ to the $i$th cluster center. That

is to say, the larger the distance is, the degree of membership of $x_j$ to the $i$th cluster center is smaller.

The parameter $m(1 < m < \infty)$ defines the level of cluster fuzziness. As the value of $m$ gets closer to 1, the cluster solution becomes increasingly similar to the solution in hard clustering techniques such as $k$-means clustering. On the other hand, when the value of $m$ approaches to infinite, the clustering solution leads to complete fuzziness.

The centroid of a cluster is defined by

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^m \cdot x_i}{\sum_{i=1}^{N} u_{ij}^m}, \tag{A.3}$$

where

- $c_j$ is the d-dimension center of the cluster.
- $u_{ij}$ is the degree to which an observation $x_i$ belongs to a cluster $c_j$.

In FCM, fuzzy partitioning is carried out through an iterative optimization of the objective function shown in equation A.1, with the update of membership $u_{ij}$ and the cluster centers $c_j$. The algorithms runs into the following steps:

1) Specify a number of clusters C.

2) Assign randomly to each point the degrees of membership for being in the clusters.

3) Repeat the iteration until the maximum number of iterations is reached, or the change of degree of membership between two iterations is no more than ε (the given sensitivity threshold). This stoping criteria is denoted as $max_{ij} | u_{ij}^{(k+1)} - u_{ij}^k | < \varepsilon$.

# Appendix B

# The theory of evidential clustering

This Appendix presents the theory of evidential *c*-means (ECM) clustering algorithm. The following basic notions and theory are taken from the research paper [MD08b].

## B.1 The evidence theory

The evidence theory [Dem67, SK94] is based on several fundamentals such as the *Basic Belief Assignment* (BBA). A BBA $m$ is the mapping from elements of the power set $2^\Theta$ on to [0, 1]:

$$m : 2^\Theta \longrightarrow [0,1],$$

where $\Theta$ is the *frame of discernment*. It is the set of possible answers for a treated problem and is composed of K exhaustive and exclusive hypotheses $\Theta = \{\omega_1, \omega_2, ..., \omega_K\}$. A BBA $m$ is written as follows:

$$\begin{cases} \sum\limits_{A \subseteq \Theta} m(A) = 1 \\ m(\emptyset) \geq 0. \end{cases} \tag{B.1}$$

Assuming that a source of information has a reliability rate equal to $(1-\alpha)$ where $(0 \leq \alpha \leq 1)$, such a meta-knowledge can be taken into account using the discounting operation introduced by[Sha76a], and is defined by:

$$\begin{cases} m^\alpha(A) = (1-\alpha) \times m(A) & \forall A \subset \Theta \\ m^\alpha(\Theta) = (1-\alpha) \times m(\Theta) + \alpha. \end{cases} \tag{B.2}$$

Discounting rates allow to correct raw data, based on learned decisions given by belief functions [ELM10]. There are several approaches for discounting, such as classical discounting of information [Sha76b], contextual discounting [MQD08], and contextual discounting with error rate [Mer10].

A discount rate α equal to 1 means that the source is not reliable and the piece of information that is provided cannot be taken into account. On the contrary, a null discount rate indicates that the source is fully reliable and the piece of information that is provided is entirely acceptable.

Within the evidence theory, several combination rules have been introduced, and this work focuses on the Dempster rule of combination [Dem67]. Assuming two BBAs $m_1$ and $m_2$ modelling two independent reliable sources of information $S_1$ and $S_2$, the Dempster rule of combination is defined as follows:

$$m = m_1 \oplus m_2, \tag{B.3}$$

so that :

$$m(\mathrm{A}) = \frac{1}{1-m(\emptyset)} \sum_{\mathrm{B}\cap\mathrm{C}=\mathrm{A}} m_1(\mathrm{B}) \times m_2(\mathrm{C}) = \frac{1}{1-m(\emptyset)} m(\mathrm{A}),$$
$$\forall \mathrm{A} \subseteq \Theta, \mathrm{A} \neq \emptyset, \tag{B.4}$$

where $m(\emptyset)$ is defined by:

$$m(\emptyset) = \sum_{\mathrm{B}\cap\mathrm{C}=\emptyset} m_1(\mathrm{B}) \times m_2(\mathrm{C}) = m(\emptyset). \tag{B.5}$$

$m(\emptyset)$ represents the conflict mass between $m_1$ and $m_2$.

The pignistic probability, denoted B$et$P, is proposed by Smets et al. [Sme05] within the Transferable Belief Model (TBM). In the decision phase, the pignistic transformation consists in distributing equiprobably the mass of a proposition A on its included hypotheses. Formally, the pignistic probability is defined by:

$$\mathrm{B}et\mathrm{P}(\omega_n) = \sum_{\mathrm{A}\subseteq\Theta} \frac{|\omega_n \cap \mathrm{A}|}{|\mathrm{A}|} \times m(\mathrm{A}), \qquad \forall \omega_n \in \Theta. \tag{B.6}$$

where || is the cardinality operator.

## B.2 Evidential *c*-means (ECM)

### B.2.1 Credal partition

In section B.1, the partial knowledge regarding the class membership of an object $i$ by a bba $m_i$ on the set $\Theta = \{\omega_1, \omega_2, ..., \omega_i\}$. This representation allows to model all situations ranging from complete ignorance to full certainty concerning the class of $i$.

A credal partition is defined as the n-tuple $M = (m_1, ..., m_n)$. It can be seen as a general model of partitioning [MD08b]:

- when each $m_i$ is a *certain* bba, then M defines a conventional, crisp partition of the set of objects. This identifies a situation of complete knowledge;
- when each $m_i$ is a *Bayesian* bba, then M specifies a fuzzy partition, as defined in [BKKP99];
- when the focal elements of all bbas are restricted to be singletons of $\Theta$ or the empty set, a partition similar to the one of [Dav96] is recovered.

A credal partition $M = (m_1, ..., m_n)$ is considered to have size $c$ if [MD08b]:

- each bba $m_i, i = 1, ..., n$ is defined on a frame $\Theta$ of $c$ elements, and
- each class has a strictly positive degree of plausibility for at least one object, i.e., for all $\omega \in \Theta$, we have $pl_i(\omega) > 0$ for some $i \in 1, ..., n$, where $pl_i$ is the plausibility function associated to $m_i$. $pl$ is defined as:

$$pl(\omega) \triangleq \sum_{B \cap A \neq \emptyset} m(B), \quad \forall A \subseteq \Theta. \tag{B.7}$$

### B.2.2 Objective function

A credal partition from object data is computed through determining, for each object $i$, the quantities $m_{ij} = m_i(A_j)(A_j \neq \emptyset, A_j \subseteq \Theta)$ in such a way that $m_{ij}$ is low (with regard to high) when the distance $d_{ij}$ between $i$ and the focal set $A_j$ is high (with regard to low). Similar to fuzzy clustering, ECM assumes that each class $\omega_k$ is represented by a center $\mathbf{v}_k \in \mathbb{R}^p$. For each subset $A_j$ of $\Theta$, the barycenter $\bar{\mathbf{v}}_j$ of the centers are associated to the classes composing $A_j$. More specifically, by introducing the notation

$$S_{kj} = \begin{cases} 1, & \text{if } \omega_k \in A_j, \\ 0, & \text{else.} \end{cases} \tag{B.8}$$

The barycenter $\bar{\mathbf{v}}_j$ associated to $A_j$ is computed by

$$\bar{\mathbf{v}}_j = \frac{1}{c_j} \sum_{k=1}^{c} S_{kj} \mathbf{v}_k, \tag{B.9}$$

where $c_j = |A_j|$ denotes the cardinal of $A_j$. The distance $d_{ij}$ is then defined by

$$d_{ij}^2 \triangleq \|\mathbf{x}_i - \bar{\mathbf{v}}_j\|^2. \tag{B.10}$$

Based on the definition of a credal partition, a separate treatment of the empty set is proposed. This particular focal element is indeed assimilated to a noise cluster, which allows to detect atypical data. Thus, an additional term for describing the credal partition for empty set is introduced in the objective function. This noise cluster the computation of the objective function depending on a fixed distance $\delta$ between all objects and the empty set.

Finally, the credal partition $M = (m_1, ..., m_d) \in \mathbb{R}^{n \times 2^c}$ and the matrix V of size (C × P) of cluster centers are computed to minimize the following objective function:

$$J_{ECM}(M,V) \triangleq \sum_{i=1}^{d} \sum_{\{j/A_j \neq \emptyset, A_j \subseteq \Theta\}} c_j^{\alpha} m_{ij}^{\beta} d_{ij}^2 + \sum_{i=1}^{n} \delta^2 m_{i\emptyset}^{\beta}, \tag{B.11}$$

subject to:

$$\sum_{\{j/A_j \neq \emptyset, A_j \subseteq \Theta\}} m_{ij} + m_{i\emptyset} = 1, \quad \forall i = 1, \ldots, d, \tag{B.12}$$

where $m_{i\emptyset}$ and $m_{ij}$ respectively denote $m_i(\emptyset)$ and $m_i(A_j)$. M is the credal partition $M = (m_1, \ldots, m_d)$ and V is a cluster centers matrix. $c_j^{\alpha}$ is a weighting coefficient and $d_{ij}$ is the Euclidean distance. In this work, the default values prescribed by the authors in [MD08a] are used, i.e. $\alpha = 1$, $\beta = 2$ and $\delta = 10$.

### B.2.3 Optimization

To minimize $J_{ECM}$, an alternate optimization scheme is designed similar to the fuzzy *c*-means (FCM) algorithm. First, the cluster centers matrix V is considered as fixed. To solve the constrained minimization problem with regard to M, the $n$ Lagrange multipliers $\lambda_i$ is introduced:

$$\mathscr{L}(\mathrm{M},\lambda_1,...,\lambda_d) = \mathrm{J_{ECM}}(\mathrm{M},\mathrm{V}) - \sum_{i=1}^{n}\lambda_i\left(\sum_{\{j/\mathrm{A}_j\neq\emptyset,\mathrm{A}_j\subseteq\Theta\}} m_{ij} + m_{i\emptyset} - 1\right). \tag{B.13}$$

Then we differentiate the Lagrangian with respect to the $m_{ij}$ , $m_{i\emptyset}$, and $\lambda_i$. After that, the derivatives are set to zero. Then it can be obtained that:

$$\frac{\partial\mathscr{L}}{\partial m_{ij}} = \beta c_j^{\alpha} m_{ij}^{\beta-1} d_{ij}^2 - \lambda_i = 0, \tag{B.14}$$

$$\frac{\partial\mathscr{L}}{\partial m_{i\emptyset}} = \beta\delta^2 m_{ij}^{\beta-1} - \lambda_i = 0, \tag{B.15}$$

$$\frac{\partial\mathscr{L}}{\partial\lambda_i} = \sum_{\{j/\mathrm{A}_j\neq\emptyset,\mathrm{A}_j\subseteq\Theta\}} m_{ij} + m_{i\emptyset} - 1 = 0. \tag{B.16}$$

From Equation B.14, it can be computed that

$$m_{ij} = \left(\frac{\lambda_i}{\beta}\right)^{1/(\beta-1)}\left(\frac{1}{c_j^{\alpha} d_{ij}^2}\right)^{1/(\beta-1)}. \tag{B.17}$$

And from Equation B.15, it can be obtained that

$$m_{i\emptyset} = \left(\frac{\lambda_i}{\beta}\right)^{1/(\beta-1)}\left(\frac{1}{\delta^2}\right)^{1/(\beta-1)}. \tag{B.18}$$

Using the Equations B.16-B.18:

$$\left(\frac{\lambda_i}{\beta}\right)^{1/(\beta-1)} = \left(\sum_j \frac{1}{c_j^{\alpha/(\beta-1)}}\frac{1}{d_{ij}^{2/(\beta-1)}} + \frac{1}{\delta^{2/(\beta-1)}}\right)^{-1}. \tag{B.19}$$

Then return to Equation B.17, the necessary condition of optimality for M can be obtained:

$$m_{ij} = \frac{c_j^{-\alpha/(\beta-1)} d_{ij}^{2/(\beta-1)}}{\sum\limits_{\mathrm{A}_k\neq\emptyset} c_k^{-\alpha/(\beta-1)} d_{ij}^{2/(\beta-1)} + \delta^{-2/(\beta-1)}}, \forall i = 1,n, \forall j/\mathrm{A}_j\subseteq\Theta, \mathrm{A}_j\neq\emptyset, \tag{B.20}$$

and

$$m_{i\emptyset} = 1 - \sum_{\mathrm{A}_j \neq \emptyset} m_{ij}, \forall i = 1, n. \tag{B.21}$$

Then, the credal partition M is considered as fixed. The minimization of $\mathrm{J_{ECM}}$ with respect to V is an unconstrained optimization problem. The partial derivatives of $\mathrm{J_{ECM}}$ with respect to the centers are given by

$$\frac{\partial \mathrm{J_{ECM}}}{\partial \mathbf{v}_l} = \sum_{i=1}^{d} \sum_{\mathrm{A}_j \neq \emptyset} c_j^{\alpha} m_{ij}^{\beta} \frac{\partial d_{ij}^2}{\partial \mathbf{v}_l}. \tag{B.22}$$

$$\frac{\partial d_{ij}^2}{\partial \mathbf{v}_l} = 2(s_{lj})(\mathbf{x}_i - \bar{\mathbf{v}}_j)\left(-\frac{1}{c_j}\right). \tag{B.23}$$

From Equation B.22 and B.23,

$$\frac{\partial \mathrm{J_{ECM}}}{\partial \mathbf{v}_l} = -2\sum_{i=1}^{d} \sum_{\mathrm{A}_j \neq \emptyset} c_j^{\alpha-1} m_{ij}^{\beta} s_{lj}(\mathbf{x}_i - \mathbf{v}_j) = -2\sum_{i=1}^{d} \sum_{\mathrm{A}_j \neq \emptyset} c_j^{\alpha-1} m_{ij}^{\beta} s_{lj}\left(\mathbf{x}_i - \frac{1}{c_j}\sum_k s_{kj}\mathbf{v}_k\right), \forall l = 1, c. \tag{B.24}$$

when setting these derivatives to zero, linear equations are given to $l$ in $\mathbf{v}_k$:

$$\sum_i \mathbf{x}_i \sum_{\mathrm{A}_j \neq \emptyset} c_j^{\alpha-1} m_{ij}^{\beta} s_{lj} = \sum_k \mathbf{v}_k \sum_i \sum_{\mathrm{A}_j \neq \emptyset} c_j^{\alpha-2} m_{ij}^{\beta} s_{lj} s_{kj}, l = 1, c. \tag{B.25}$$

Let B be a matrix of size $(c \times p)$, which is defined by

$$\mathrm{B}_{lq} = \sum_{i}^{d} x_{iq} \sum_{\mathrm{A}_j \neq \emptyset} c_j^{\alpha-1} m_{ij}^{\beta} s_{lj} = \sum_{i}^{d} x_{iq} \sum_{\mathrm{A}_j \ni \omega_l} c_j^{\alpha-1} m_{ij}^{\beta}, l = 1, c, q = 1, p, \tag{B.26}$$

and let H be a matrix of size $(c \times c)$ given by

$$\mathrm{H}_{lk} = \sum_i \sum_{\mathrm{A}_j \neq \emptyset} c_j^{\alpha-2} m_{ij}^{\beta} s_{lj} s_{kj} = \sum_i \sum_{\mathrm{A}_j \supseteq \{\omega_k, \omega_k\}} c_j^{\alpha-2} m_{ij}^{\beta}, k, l = 1, c. \tag{B.27}$$

Based on the above notions, V is solution of the following linear system:

$$\mathrm{HV} = \mathrm{B}, \tag{B.28}$$

which can be solved using a standard linear system solver.

# Appendix C

# The fast non-dominated sorting algorithm

This Appendix gives the theory of multi-objective optimization and the fast non-dominated sorting algorithm. The theory is taken from research papers [OFGC17] and [Mie12].

## C.1 Multi-objective optimization

Assume having $v \geqslant 2$ conflicting objectives, described by functions $f_1(x), f_2(x),...,f_v(x)$, where $x = (x_1, x_2, ..., x_n)$ is a vector of variables (decision vector) and $n$ is the number of variables or dimension of the problem. A multi-objective minimization problem is formulated as follows [Mie12]:

$$\underset{x \in \mathrm{S}}{min} f(x) = [f_1(x), f_2(x), ..., f_v(x)]^{\mathrm{T}}, \tag{C.1}$$

where $z = f(x)$ is an *objective vector*, defining the values for all objective functions, $f_i : \mathrm{S} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$, $i \in \{1, 2, ..., v\}$, where $v$ is the number of objective functions, and $\mathrm{S} \subseteq \mathbb{R}^n$ is an $n$-dimensional euclidean space, which defines all feasible decision vectors. The feasible set is typically defined by some constraint functions. If some objective function is to be maximized, it is equivalent to minimize its negative. Typically, in multi-objective optimization, there exists no solution for minimizing all objective functions simultaneously. Therefore, Pareto optimal solutions are considered and optimization results are provided to the decision maker.

In mathematical terms, a feasible solution $x' \in \mathrm{S}$ is a *Pareto optimal solution*

that Pareto dominates another solution $x \in S$ if $f_i(x') \leqslant f_i(x)$ for all $x \in S$ and $f_j(x') < f_j(x)$ for at most one $j$. Objective vectors are defined as optimal if none of their elements can be further improved without worsen at least one of the other elements. An objective vector $z' = f(x')$ is Pareto optimal if the corresponding decision vector $x'$ is Pareto optimal. The set of all the Pareto optimal decision vectors is called the *Pareto set*. The region defined by all the objective function values for the Pareto set points is called the *Pareto front* [OFGC17].

An objective vector $z' \in \mathbb{R}^v$ dominates another objective vector $z \in \mathbb{R}^v$ (or $z' \succ z$) if $z' \leqslant z$ for all $i = 1, ..., v$ and there exists at least one $j$ such that $z'_j < z_j$.

In multi-objective optimization algorithms, the subset of solutions in a population whose objective vectors are not dominated by any other objective vector is called the *non-dominated set*, and the objective vectors are called the *non-dominated objective vectors*. The algorithms involve more than one objective function that are to be minimized or maximized, and the generated solution is set of solutions that define the best trade-off between competing objectives. The main aim of the multi-objective optimization algorithms is to generate well-distributed non-dominated objective vectors as close as possible to the Pareto front [OFGC17].

## C.2 Fast non-dominated sorting

The fast non-dominated sorting algorithm assigns ranks to the individuals, and classifies the population into several non-dominated levels (so-called fronts) (Fig. C.1). The population is sorted into a hierarchy of sub-populations based on the ordering of Pareto dominance. According to Pareto dominance, the individuals with the same rank are considered non-dominated among each other. These individuals can only be dominated by the solutions in a lower rank. Dominance comparisons between the individuals is the most repeated operation in non-dominated sorting with a high computational burden, which determines the algorithm efficiency [OFGC17].

Algorithm 6 (FNDS-UDC) shows the pseudocode of the fast non-dominated sorting algorithm. The presented algorithm can be divided into three phases [OFGC17]: *Phase 1-Unidirectional Dominance Comparison*, where comparisons between individuals of initial population P are computed and the information about their dominance is collected. *Phase 2*, where the individuals of the first front are classified, and *Phase 3-Front assignment*, where the initial population P is classified in several fronts by the usage of dominance information. The algorithm sorts

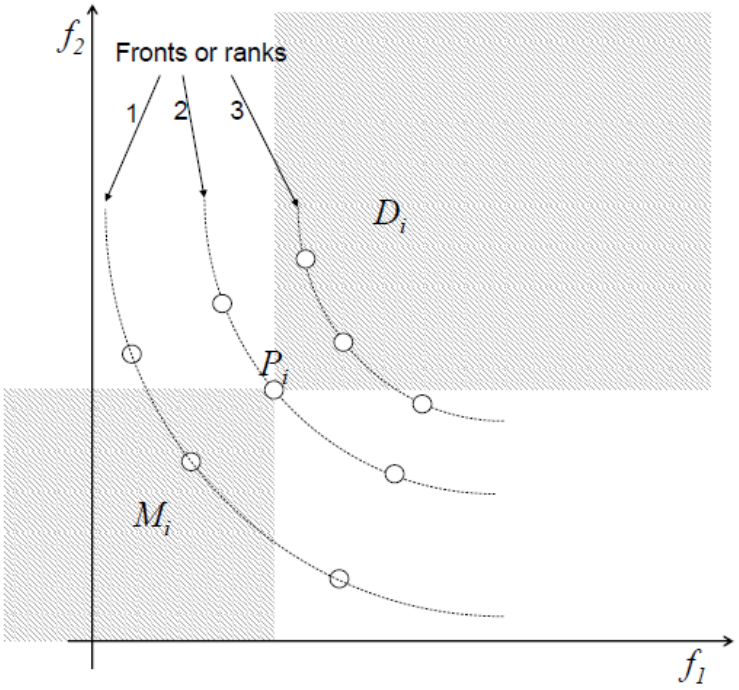



Figure C.1 – Pareto fronts and dominance among individuals in a two dimensional space [OFGC17].

the population (P) in ranks (or fronts) (R$i$) starting from the front 1. The following notations are used [OFGC17]:

- $n$: the dimension of the search region S $\in \mathbb{R}^n$.
- $p$: the number of individuals.
- $v$: the number of objective functions.
- $\mathrm{X}_{(p\times n)}$: the matrix of individuals' decision vectors.
- $\mathrm{V}_{(p\times v)}$: the matrix of the individuals' objective vectors.
- $rank(\mathrm{P}_i)$: the rank of $\mathrm{P}_i$.
- $\mathrm{T}_{(p\times 1)}$: the column vector with the rank of each individual, i.e., $\mathrm{T}_i = rank(\mathrm{P}_i)$, $i = 1, ..., p$.
- $\mathrm{P} = [\mathrm{X}|\mathrm{V}|\mathrm{T}]_{(\mathrm{P}\times(n+v+1))}$: the matrix of individuals, as the result of the concatenation of the columns of X, V and T.

- $F_{(p\times p)}$: the front matrix. It stores the individuals at each front or rank. Each row $F_i$ of F stores the set of indices of individuals at front $i$, i.e., $F_i = \{j : rank(P_j) = i\}$.

- $\#F_{(p\times 1)}$: the column vector with the number of individuals at each front. $\#F_i = |\{j : rank(P_j) = i\}|$.

- $t_max$: the number of fronts, i.e. the number of non zero rows of #F.

- $R = [\#F|F]_{(p\times(p+1))}$: the rank matrix, as the result of the concatenation of the columns of #F and F.

- $\#M_{(p\times 1)}$: the column vector with the number of dominators (masters) of each individual.

- $\#D_{(p\times 1)}$: the column vector with $D_i = |\{j : P_j \prec P_i\}|$, i.e, the element $i$ stores the number of individuals dominated by $P_i$.

- $D_{(p\times p)}$: the dominated matrix. The row $i$ stores the indexes of individuals dominated by $P_i$, i.e., $D_i = \{j : P_j \prec P_i\}$.

- $MD = [\#M|\#D|D]_{(p\times(p+2))}$: the masters-dominated matrix, as the result of the concatenation of the columns of #M, #D, and D.

Each $i^{th}$ iteration of the FNDS-UDC algorithm is unique because it contains conditional instructions that can change the program trace. When assuming the population to be sorted by the algorithm is large enough, the workload for several sets of iterations can be considered to be near to their average value [OFGC17].

The input of the FNDS-UDC is the population P and the output is the rank matrix R and the update of T in the P structure. As a result, FNDS-UDC generates the dominance matrix D which is very sparse to calculate R. P, MD and R matrices have been defined as static data structures of size $(p \times (n + v + 1))$, $(p \times (p + 2))$, and $(p \times (p + 1))$, respectively. It means that the sizes of these matrices are invariant throughout their lifetime, and they are set to their maximum size. Because the number of fronts (or ranks) is unknown when FNDS-UDC starts, the number of rows of R is set to the number of individuals $p$. When considering P and MD, these two matrices have been stored column-wise, which presents a better computation consumption than the row-wise approach.

The advantage of the FNDS-UDC algorithm is its low computation memory requirement. On one hand, the dominance checking between every individual and the population is computed with a high spatial locality in the access to P. On the other hand, first columns of MD are frequently acceded to update the number of dominators and dominated of an individual [OFGC17]. In a CPU or GPU, the algorithm sorts MD in column order. This increases the probability of maintaining first columns in cache memory and the number of cache misses is consequently reduced. This sorting strategy has significantly improves the performance of FNDS-UDC when the population is very large, since most of the running time consumed by the algorithm is due to the memory access performed in the dominance comparison [OFGC17].

Compared to traditional non-dominated sorting algorithms, the FNDS-UDC algorithm overcomes their drawback of oversizing MD and R due to the unknown dominance patterns (for guaranteeing enough memory for them). However, the memory requirements for R could be greatly reduced if the number of fronts is a priori known [OFGC17].

**Algorithm 6** Fast non-dominated sorting procedure - Unidirectional Dominance Comparison (FNDS-UDC) [OFGC17].

**Input:**

$n$: the dimension of the search space.
$p$:the number of individuals.
$v$:the number of objective functions.
$\mathrm{P}_{(p\times(n+v+1))}$: population.
$\#\mathrm{M}_i$: the column vector with the number of dominators ($i$) of each individual.
$\#\mathrm{D}_i$: the column vector that contains the element $i$.

**Output:**

F: the front matrix.
T: the rank of each individual.
**Phase 1: Unidirectional Dominance Comparison (UDC)**
**for** $i \leftarrow 1$ **to** $p$ **do**
    $\mathrm{D}_i \leftarrow \{\emptyset\}$
    $\#\mathrm{M}_i \leftarrow 0$
    **for** $j \leftarrow 1$ **to** $p$ **do**
        **for** $k \leftarrow 1$ **to** $v$ **do**
        ▷ Check dominance between individuals $\mathrm{P}_i$ and $\mathrm{P}_j$ for the objective $k$.
        **if** $\mathrm{P}_i \succ \mathrm{P}_j$ **then**
            $\mathrm{D}_i \succ \mathrm{D}_i \cup \{j\}$
                ▷ Add $\mathrm{P}_j$ to the set of individuals dominated by $\mathrm{P}_i$.
        **else if** $\mathrm{P}_j \succ \mathrm{P}_i$ **then**
            $\#\mathrm{M}_i \succ \#\mathrm{M}_i + 1$
                ▷ Increase dominators counter for $\mathrm{P}_i$.
**Phase 2: Assign front to the individuals of the $1_{st}$ front**
**for** $i \leftarrow 1$ **to** $p$ **do**
    **if** $\#\mathrm{M}_i = 0$ **then** ▷ If individual $\mathrm{P}_i$ belongs to the $1_{st}$ front.
        $\mathrm{T}_i = 1$ ▷ Set individual $i$ to the $1_{st}$ front.

        $\mathrm{F}_1 \leftarrow \mathrm{F}_1 \cup \{i\}$
**Phase 3: Front Assignment**
$t = 1$
**while** $\mathrm{F}_t \neq \{\emptyset\}$ **do**
    $\mathrm{Q} \leftarrow \{\emptyset\}$ ▷ Temporary set used to store individuals of the next front.
    **for each** $i \in \mathrm{F}_t$ **do**
        **for each** $j \in \mathrm{D}_i$ **do**
            $\#\mathrm{M}_i \leftarrow \#\mathrm{M}_i - 1$
            **if** $\#\mathrm{M}_i = 0$ **then**
                $\mathrm{T}_i \leftarrow t + 1$ ▷ Set individual $j$ to the $t+1$ front.
                $\mathrm{Q} \leftarrow \mathrm{Q} \cup \{j\}$
    $t \leftarrow t + 1$ ▷ Increase the front counter.
    $\mathrm{F}_t \leftarrow \mathrm{Q}$
$t_{max} \leftarrow t$
**return** F, T ▷ Returns the Pareto Dominance Ranking and the Front Matrix.

# Bibliography


[Abb12] Sunitha Abburu. A survey on ontology reasoners and comparison. International Journal of Computer Applications, 57(17), 2012.

[Abr05] Ajith Abraham. Rule-based expert systems. Handbook of measuring system design, 2005.

[AC99] Aijun An and Nick Cercone. An empirical study on rule quality measures. In International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing, pages 482–491. Springer, 1999.

[AC01] Aijun An and Nick Cercone. Rule quality measures for rule induction systems: Description and evaluation. Computational Intelligence, 17(3):409–424, 2001.

[AC+10] Sylvain Arlot, Alain Celisse, et al. A survey of cross-validation procedures for model selection. Statistics surveys, 4:40–79, 2010.

[AD06] Farhad Ameri and Debasish Dutta. An upper ontology for manufacturing service description. In ASME 2006 international design engineering technical conferences and computers and information in engineering conference, pages 651–661. American Society of Mechanical Engineers Digital Collection, 2006.

[AD08] Farhad Ameri and Debasish Dutta. An upper ontology for manufacturing service description. In ASME 2006 international design engineering technical conferences and computers and information in engineering conference, pages 651–661. American Society of Mechanical Engineers Digital Collection, 2008.

[AGN19] Fazel Ansari, Robert Glawar, and Tanja Nemeth. Prima: a prescriptive maintenance model for cyber-physical production systems. International Journal of Computer Integrated Manufacturing, pages 1–22, 2019.

[ALRL04] Algirdas Avizienis, J-C Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. IEEE transactions on dependable and secure computing, 1(1):11–33, 2004.

[AM14] Farhad Ameri and Christian McArthur. Semantic rule modelling for intelligent supplier discovery. International Journal of Computer Integrated Manufacturing, 27(6):570–590, 2014.

[ANAH08] Mouhib Al-Noukari and Wael Al-Hussan. Using data mining techniques for predicting future car market demand; dcx case study. In 2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications, pages 1–5. IEEE, 2008.

[AP94] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications, 7(1):39–59, 1994.

[ARS97] Khaled Alsabti, Sanjay Ranka, and Vineet Singh. An efficient k-means clustering algorithm. 1997.

[AS+94] Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB, volume 1215, pages 487–499, 1994.

[AS+95] Rakesh Agrawal, Ramakrishnan Srikant, et al. Mining sequential patterns. In icde, volume 95, pages 3–14, 1995.

[AUM12] Farhad Ameri, Colin Urbanovsky, and Christian McArthur. A systematic approach to developing ontologies for manufacturing service modeling. In Proceedings of the workshop on ontology and semantic web for manufacturing, page 14. Citeseer, 2012.

[Aya18] Ali Ayadi. Semantic approaches for the meta-optimization of complex biomolecular networks. PhD thesis, 2018.

[B+79] Avron Barr et al. Meta-knowledge and cognition. In IJCAI, pages 31–33, 1979.

[BAB+03] Ralph Bergmann, Klaus-Dieter Althoff, Sean Breen, Mehmet Göker, Michel Manago, Ralph Traphöner, and Stefan Wess. Developing industrial case-based reasoning applications: The INRECA methodology. Springer Science & Business Media, 2003.

[BBH+10] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. A survey of context modelling and reasoning techniques. Pervasive and Mobile Computing, 6(2):161–180, 2010.

[BD99] Kristin P Bennett and Ayhan Demiriz. Semi-supervised support vector machines. In Advances in Neural Information processing systems, pages 368–374, 1999.

[BDPH06] Ladjel Bellatreche, Nguyen Xuan Dung, Guy Pierra, and Dehainsala Hondjack. Contribution of ontology-based data modeling to automatic integration of electronic catalogues within engineering databases. Computers in Industry, 57(8-9):711–724, 2006.

[BDR07] Matthias Baldauf, Schahram Dustdar, and Florian Rosenberg. A survey on context-aware systems. International Journal of Ad Hoc and Ubiquitous Computing, 2(4):263–277, 2007.

[BDRC10] Mouloud Boumahdi, Jean-Paul Dron, Saïd Rechak, and Olivier Cousinard. On the extraction of rules in the identification of bearing defects in rotating machinery using decision tree. Expert Systems with Applications, 37(8):5887–5894, 2010.

[BEC+10] Milton Borsato, Carla Cristina Amodio Estorilio, Carlos Cziulik, Cassia Maria Lie Ugaya, Henrique Rozenfeld, et al. An ontology building approach for knowledge sharing in product lifecycle management. International Journal of Business and Systems Research, 4(3):278, 2010.

[BEF84] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. Computers & Geosciences, 10(2-3):191–203, 1984.

[Ber88] John A Bernard. Use of a rule-based system for process control. IEEE Control Systems Magazine, 8(5):3–13, 1988.

[BGR02] Matthew J Beal, Zoubin Ghahramani, and Carl E Rasmussen. The infinite hidden markov model. In Advances in neural information processing systems, pages 577–584, 2002.

[BHGS01] Sean Bechhofer, Ian Horrocks, Carole Goble, and Robert Stevens. Oiled: a reason-able ontology editor for the semantic web. In Annual Conference on Artificial Intelligence, pages 396–408. Springer, 2001.

[BKKP99] James C Bezdek, James Keller, Raghu Krisnapuram, and Nikhil Pal. Fuzzy models and algorithms for pattern recognition and image processing, volume 4. Springer Science & Business Media, 1999.

[BL07] Stefano Borgo and Paulo Leitão. Foundations for a core ontology of manufacturing. In Ontologies, pages 751–775. Springer, 2007.

[BPERFM09] Enrique Bonsón-Ponte, Tomás Escobar-Rodríguez, and Francisco Flores-Muñoz. Towards an ontology-based network for banking supervision. Online Information Review, 33(5):943–955, 2009.

[Bro09] Paul Browne. JBoss Drools business rules. Packt Publishing Ltd, 2009.

[Bry86] Randal E Bryant. Graph-based algorithms for boolean function manipulation. Computers, IEEE Transactions on, 100(8):677–691, 1986.

[BS04] Steffen Bickel and Tobias Scheffer. Multi-view clustering. In ICDM, volume 4, pages 19–26, 2004.

[BS11] Fernando Bobillo and Umberto Straccia. Fuzzy ontology representation using owl 2. International Journal of Approximate Reasoning, 52(7):1073–1094, 2011.

[BT97] Riza C Berkan and Sheldon Trubatch. Fuzzy system design principles. Wiley-IEEE Press, 1997.

[CAS+09] Alvaro Cardenas, Saurabh Amin, Bruno Sinopoli, Annarita Giani, Adrian Perrig, Shankar Sastry, et al. Challenges for securing cyber physical systems. In Workshop on future directions in cyber-physical systems security, volume 5, 2009.

[CBB+12] Michael Compton, Payam Barnaghi, Luis Bermudez, RaúL GarcíA-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, et al. The ssn ontology of the w3c semantic sensor network incubator group. Web semantics: science, services and agents on the World Wide Web, 17:25–32, 2012.

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3):15, 2009.

[CBS+17] Federico Civerchia, Stefano Bocchino, Claudio Salvadori, Enrico Rossi, Luca Maggiani, and Matteo Petracca. Industrial internet of things monitoring solution for advanced predictive maintenance applications. Journal of Industrial Information Integration, 7:4–12, 2017.

[CGZM+19] Qiushi Cao, Franco Giustozzi, Cecilia Zanni-Merk, François de Bertrand de Beuvron, and Christoph Reich. Smart condition monitoring for industry 4.0 manufacturing processes: An ontology-based approach. Cybernetics and Systems, 50(2):82–96, 2019.

[CMM12] Damien Cram, Benoît Mathern, and Alain Mille. A complete chronicle discovery approach: application to activity analysis. Expert Systems, 29(4):321–346, 2012.

[CP05] Se-Hak Chun and Yoon-Joo Park. Dynamic adaptive ensemble case-based reasoning: application to stock market prediction. Expert Systems with Applications, 28(3):435–443, 2005.

[CP09] Shunfeng Cheng and Michael Pecht. A fusion prognostics method for remaining useful life prediction of electronic products. In 2009 IEEE International Conference on Automation Science and Engineering, pages 102–107. IEEE, 2009.

[CS03] Silvia Coradeschi and Alessandro Saffiotti. An introduction to the anchoring problem. Robotics and Autonomous Systems, 43(2-3):85–96, 2003.

[CSB06] Song Hui Chon, Malcolm Slaney, and Jonathan Berger. Predicting success from music sales data: a statistical and adaptive approach. In Proceedings of the 1st ACM workshop on Audio and music computing multimedia, pages 83–88. ACM, 2006.

[CSSG11] Jose Celaya, Abhinav Saxena, Sankalita Saha, and Kai F Goebel. Prognostics of power mosfets under thermal stress accelerated aging using data-driven and model-based methodologies. 2011.

[CWAM75] Allan Collins, Eleanor H Warnock, Nelleke Aiello, and Mark L Miller. Reasoning from incomplete knowledge. In Representation and understanding, pages 383–415. Elsevier, 1975.

[CZMR18a] Qiushi Cao, Cecilia Zanni-Merk, and Christoph Reich. Ontologies for manufacturing process modeling: A survey. In International Conference on Sustainable Design and Manufacturing, pages 61–70. Springer, 2018.

[CZMR18b] Qiushi Cao, Cecilia Zanni-Merk, and Christoph Reich. Towards an ontological representation of condition monitoring knowledge in the manufacturing domain. In KEOD, pages 310–316, 2018.

[CZMR19] Qiushi Cao, Cecilia Zanni-Merk, and Christoph Reich. Towards a core ontology for condition monitoring. Procedia Manufacturing, 28:177–182, 2019.

[DAS01] Anind K Dey, Gregory D Abowd, and Daniel Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human–Computer Interaction, 16(2-4):97–166, 2001.

[Dav96] Rajesh N Dave. Validating fuzzy partitions obtained through c-shells clustering. Pattern recognition letters, 17(6):613–623, 1996.

[DBRA+11] Daniele Di Bona, Giuseppe Lo Re, Giovanni Aiello, Adriano Tamburo, and Marco Alessi. A methodology for graphical modeling of business rules. In 2011 UKSim 5th European Symposium on Computer Modeling and Simulation, pages 102–106. IEEE, 2011.

[DD99] Christophe Dousson and Thang Vu Duong. Discovering chronicles with numerical time constraints from alarm logs for monitoring dynamic systems. In IJCAI, volume 99, pages 620–626, 1999.

[DDVvHB17] Daniele Dell'Aglio, Emanuele Della Valle, Frank van Harmelen, and Abraham Bernstein. Stream reasoning: A survey and outlook. Data Science, 1(1-2):59–83, 2017.

[Dem67] A.P. Dempster. Upper and lower probabilities induced by multivalued mapping. AMS-38, 1967.

[DFH11] John Domingue, Dieter Fensel, and James A Hendler. Handbook of semantic web technologies. Springer Science & Business Media, 2011.

[DGD+09] Otilia Elena Dragomir, Rafael Gouriveau, Florin Dragomir, Eugenia Minca, and Noureddine Zerhouni. Review of prognostic problem in condition-based maintenance. In 2009 European Control Conference (ECC), pages 1587–1592. IEEE, 2009.

[DLIPBRS03] Beatriz De La Iglesia, Mark S Philpott, Anthony J Bagnall, and Vic J Rayward-Smith. Data mining rules using multi-objective evolutionary algorithms. In The 2003 Congress on Evolutionary Computation, 2003. CEC'03., volume 3, pages 1552–1559. IEEE, 2003.

[dlIRRS05] Beatriz de la Iglesia, Alan Reynolds, and Vic J Rayward-Smith. Developments on a multi-objective metaheuristic (momh) algorithm for finding interesting sets of classification rules. In International Conference on Evolutionary Multi-Criterion Optimization, pages 826–840. Springer, 2005.

[dSSS09] Mathieu d'Aquin, Anne Schlicht, Heiner Stuckenschmidt, and Marta Sabou. Criteria and evaluation for ontology modularization techniques. In Modular ontologies, pages 67–89. Springer, 2009.

[DWL15] Dejing Dou, Hao Wang, and Haishan Liu. Semantic data mining: A survey of ontology-based approaches. In Proceedings of the 2015 IEEE 9th international conference on semantic computing (IEEE ICSC 2015), pages 244–251. IEEE, 2015.

[DXHL14] Li Da Xu, Wu He, and Shancang Li. Internet of things in industries: A survey. IEEE Transactions on industrial informatics, 10(4):2233–2243, 2014.

[DZWL16] LY Ding, BT Zhong, Song Wu, and HB Luo. Construction risk knowledge management in bim using ontology and semantic web technology. Safety science, 87:202–213, 2016.

[EFJ+10] Christos Emmanouilidis, Luca Fumagalli, Erkki Jantunen, Petros Pistofidis, Marco Macchi, Marco Garetti, et al. Condition monitoring based on incremental learning and domain ontology for condition-based maintenance. In 11th International Conference on Advances in Production Management Systems, APMS, 2010.

[ELM10] Zied Elouedi, Eric Lefèvre, and David Mercier. Discountings of a belief function using a confusion matrix. In 2010 22nd IEEE International Conference on Tools with Artificial Intelligence, volume 1, pages 287–294. IEEE, 2010.

[EPMC12] K Efthymiou, N Papakostas, D Mourtzis, and G Chryssolouris. On a predictive maintenance platform for production systems. Procedia CIRP, 3:221–226, 2012.

[Fen01] Dieter Fensel. Ontologies. In Ontologies, pages 11–18. Springer, 2001.

[FFR97] Adam Farquhar, Richard Fikes, and James Rice. The ontolingua server: A tool for collaborative ontology construction. International journal of human-computer studies, 46(6):707–727, 1997.

[FH+08] Ernest Friedman-Hill et al. Jess, the rule engine for the java platform, 2008.

[FLGPJ97] Mariano Fernández-López, Asunción Gómez-Pérez, and Natalia Juristo. Methontology: from ontological art towards ontological engineering. 1997.

[FVLK+17] Philippe Fournier-Viger, Jerry Chun-Wei Lin, Rage Uday Kiran, Yun Sing Koh, and Rincy Thomas. A survey of sequential pattern mining. Data Science and Pattern Recognition, 1(1):54–77, 2017.

[GCMG13] Antonio Gomariz, Manuel Campos, Roque Marin, and Bart Goethals. Clasp: An efficient algorithm for mining frequent closed sequences. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 50–61. Springer, 2013.

[GDBR02] Antoine Grall, Laurence Dieulle, Christophe Bérenguer, and Michel Roussignol. Continuous-time predictive-maintenance scheduling for a deteriorating system. IEEE transactions on reliability, 51(2):141–150, 2002.

[GE03] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. Journal of machine learning research, 3(Mar):1157–1182, 2003.

[GF94] Michael Gruninger and Mark S Fox. The design and evaluation of ontologies for enterprise engineering. In Workshop on Implemented Ontologies, European Conference on Artificial Intelligence (ECAI). Citeseer, 1994.

[GGB+02] Fredrik Gustafsson, Fredrik Gunnarsson, Niclas Bergman, Urban Forssell, Jonas Jansson, Rickard Karlsson, and P-J Nordlund. Particle filters for positioning, navigation, and tracking. IEEE Transactions on signal processing, 50(2):425–437, 2002.

[GM03] Michael Gruninger and Christopher Menzel. The process specification language (psl) theory and applications. AI magazine, 24(3):63–63, 2003.

[GMF+03] John H Gennari, Mark A Musen, Ray W Fergerson, William E Grosso, Monica Crubézy, Henrik Eriksson, Natalya F Noy, and Samson W Tu. The evolution of protégé: an environment for knowledge-based systems development. International Journal of Human-computer studies, 58(1):89–123, 2003.

[Gri09] Stephan Grimm. Semantic matchmaking with nonmonotonic description logics, volume 1. IOS Press, 2009.

[Gru93] Thomas R Gruber. A translation approach to portable ontology specifications. Knowledge acquisition, 5(2):199–220, 1993.

[Gru09] Tom Gruber. Ontology. Springer, 2009.

[GRV09] Ashok K Goel, Spencer Rugaber, and Swaroop Vattam. Structure, behavior, and function of complex systems: The structure, behavior, and function modeling language. Ai Edam, 23(1):23–35, 2009.

[GW10] Giancarlo Guizzardi and Gerd Wagner. Using the unified foundational ontology (ufo) as a foundation for general conceptual modeling languages. In Theory and Applications of Ontology: Computer Applications, pages 175–196. Springer, 2010.

[GWPZ04] Tao Gu, Xiao Hang Wang, Hung Keng Pung, and Da Qing Zhang. An ontology-based context model in intelligent environments. In Proceedings of communication networks and distributed systems modeling and simulation conference, volume 2004, pages 270–275. San Diego, CA, USA., 2004.

[Ham94] James Douglas Hamilton. Time series analysis, volume 2. Princeton university press Princeton, NJ, 1994.

[HB11] Matthew Horridge and Sean Bechhofer. The owl api: A java api for owl ontologies. Semantic web, 2(1):11–21, 2011.

[HBSP05] Alec Holt, Isabelle Bichindaritz, Rainer Schmidt, and Petra Perner. Medical applications in case-based reasoning. The Knowledge Engineering Review, 20(3):289–292, 2005.

[HH00] Jeff Heflin and James Hendler. Semantic interoperability on the web. Technical report, MARYLAND UNIV COLLEGE PARK DEPT OF COMPUTER SCIENCE, 2000.

[HJM+09] Matthew Horridge, Simon Jupp, Georgina Moulton, Alan Rector, Robert Stevens, and Chris Wroe. A practical guide to building owl ontologies using protégé 4 and co-ode tools edition1. 2. The university of Manchester, 107, 2009.

[HP06] Jerry R Hobbs and Feng Pan. Time ontology in owl. W3C working draft, 27:133, 2006.

[HPSB+04] Ian Horrocks, Peter F Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, Mike Dean, et al. Swrl: A semantic web rule language combining owl and ruleml. W3C Member submission, 21(79):1–31, 2004.

[Hui98] Christian Huitema. IPv6: the new Internet protocol. Prentice Hall PTR, 1998.

[HY06] Yu Hirate and Hayato Yamana. Generalized sequential pattern mining with item intervals. JCP, 1(3):51–60, 2006.

[HZTM09] Aiwina Heng, Sheng Zhang, Andy CC Tan, and Joseph Mathew. Rotating machinery prognostics: State of the art, challenges and opportunities. Mechanical systems and signal processing, 23(3):724–739, 2009.

[IN04] Hisao Ishibuchi and Satoshi Namba. Evolutionary multiobjective knowledge extraction for high-dimensional pattern classification problems. In International Conference on Parallel Problem Solving from Nature, pages 1123–1132. Springer, 2004.

[ISO00] BSEN ISO. 9001: 2008 quality management systems. requirements. International Organization for Standardization, 2000.

[ISO04] IS ISO13372. Condition monitoring and diagnostics of machines—vocabular, 2004.

[ISO05] ISO9000 ISO. 9000-quality management systems–fundamentals and vocabulary. Quality management systems, Berlin: Beuth Verlag GmbH, pages 22–25, 2005.

[ISO07] ISO ISO. 11783-tractors and machinery for agriculture and forestry-serial control and communications data network-part 3: Data link layer, 2007.

[ISO12] ISO. Condition monitoring and diagnostics of machines–vocabulary, 2012.

[JBCV98] Dean Jones, Trevor Bench-Capon, and Pepijn Visser. Methodologies for ontology development. 1998.

[Jen13] Apache Jena. Apache jena. jena. apache. org [Online]. Available: http://jena. apache. org [Accessed: Mar. 20, 2014], page 14, 2013.

[Jen15] Apache Jena. A free and open source java framework for building semantic web and linked data applications. Available online: jena. apache. org/(accessed on 28 April 2015), 2015.

[JGZ17] Kamran Javed, Rafael Gouriveau, and Noureddine Zerhouni. State of the art and taxonomy of prognostics approaches, trends of prognostics applications and open issues towards maturity at different technology readiness levels. Mechanical Systems and Signal Processing, 94:214–236, 2017.

[JGZZ11] Kamran Javed, Rafael Gouriveau, Ryad Zemouri, and Noureddine Zerhouni. Improving data-driven prognostics by assessing predictability of features. 2011.

[JJ90] Peter Jackson and Peter Jackson. Introduction to expert systems, volume 2. Addison-Wesley Reading, MA, 1990.

[JKK+10] Anthony D JoSEP, RAnDy KAtz, AnDy KonWinSKi, LEE Gunho, DAViD PAttERSon, and ARiEL RABKin. A view of cloud computing. Communications of the ACM, 53(4), 2010.

[JMM96] Anil K Jain, Jianchang Mao, and K Moidin Mohiuddin. Artificial neural networks: A tutorial. Computer, 29(3):31–44, 1996.

[JSHL19] Eeva Järvenpää, Niko Siltala, Otto Hylli, and Minna Lanz. The development of an ontology for describing the capabilities of manufacturing resources. Journal of Intelligent Manufacturing, 30(2):959–978, 2019.

[Kal13] Gopinath Kallianpur. Stochastic filtering theory, volume 13. Springer Science & Business Media, 2013.

[KB91] Igor Kononenko and Ivan Bratko. Information-based evaluation criterion for classifier's performance. Machine Learning, 6(1):67–80, 1991.

[KCH+02] Paul Kogut, Stephen Cranefield, Lewis Hart, Mark Dutra, Kenneth Baclawski, Mieczyslaw Kokar, and Jeffrey Smith. Uml for ontology development. The Knowledge Engineering Review, 17(1):61–64, 2002.

[KCP08] Wooju Kim, Dae Woo Choi, and Sangun Park. Agent based intelligent search framework for product information using ontology mapping. Journal of Intelligent Information Systems, 30(3):227–247, 2008.

[KDG+02] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. Logistic regression. Springer, 2002.

[KITT17] Mallikarjun Kande, Alf J Isaksson, Rajeev Thottappillil, and Nathaniel Taylor. Rotating electrical machine condition monitoring automation—a review. Machines, 5(4):24, 2017.

[KM01] Agnes Kaposi and Margaret Myers. Systems for all. World Scientific, 2001.

[KMRW14] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In Advances in neural information processing systems, pages 3581–3589, 2014.

[Koh97] Teuvo Kohonen. Exploration of very large databases by self-organizing maps. In Proceedings of International Conference on Neural Networks (ICNN'97), volume 1, pages PL1–PL6. IEEE, 1997.

[KOK+17] Gabriel Krummenacher, Cheng Soon Ong, Stefan Koller, Seijin Kobayashi, and Joachim M Buhmann. Wheel defect detection with machine learning. IEEE Transactions on Intelligent Transportation Systems, 19(4):1176–1187, 2017.

[Kol14] Janet Kolodner. Case-based reasoning. Morgan Kaufmann, 2014.

[KS11] Bartosz Kucharski and Edward Szczerbicki. An approach to smart experience management. Cybernetics and Systems: An International Journal, 42(2):156–164, 2011.

[KSH12] Markus Krötzsch, Frantisek Simancik, and Ian Horrocks. A description logic primer. arXiv preprint arXiv:1201.4089, 2012.

[LBK14] Jay Lee, Behrad Bagheri, and Hung-An Kao. Recent advances and trends of cyber-physical systems and big data analytics in industrial informatics. In International proceeding of int conference on industrial informatics (INDIN), pages 1–6, 2014.

[LBK15] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manufacturing letters, 3:18–23, 2015.

[LBS13] Shuming Liu, Christopher Brewster, and Duncan Shaw. Ontologies for crisis management: A review of state of the art in ontology design and usability. In ISCRAM, pages 1–10, 2013.

[Lei10] Frank Leistner. Mastering organizational knowledge flow. Wiley Online Library, 2010.

[LFK+14] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. Business & information systems engineering, 6(4):239–242, 2014.

[LH07] Hsiao-Kang Lin and Jenny A Harding. A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration. Computers in Industry, 58(5):428–437, 2007.

[LHT17] Mohamed Layouni, Mohamed Salah Hamdi, and Sofiène Tahar. Detection and sizing of metal-loss defects in oil and gas pipelines

using pattern-adapted wavelets and machine learning. Applied Soft Computing, 52:247–261, 2017.

[LPNG13] Yan Lu, Hervé Panetto, Yihua Ni, and Xinjian Gu. Ontology alignment for networked enterprise information system interoperability in supply chain environment. International Journal of Computer Integrated Manufacturing, 26(1-2):140–151, 2013.

[LSD+] S Lemaignan, A Siadat, JY Dantan, A Semenenko, and A Mason. A proposal for an ontology of manufacturing domain. In IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications, pages 195–200.

[LSDS06] Severin Lemaignan, Ali Siadat, J-Y Dantan, and Anatoli Semenenko. Mason: A proposal for an ontology of manufacturing domain. In IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06), pages 195–200. IEEE, 2006.

[LVV03] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering algorithm. Pattern recognition, 36(2):451–461, 2003.

[LW+02] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. R news, 2(3):18–22, 2002.

[MAP18] Maribel Mendonça, Jose Aguilar, and Niriaska Perozo. Application of category theory. Ingenierie des Systemes d'Information, 23(2):11, 2018.

[MAVM11] Maxime Monnin, Bouthaina Abichou, Alexandre Voisin, and Christophe Mozzati. Fleet historical cases for predictive maintenance. In The International Conference Surveillance, volume 6, pages 25–26, 2011.

[MBB+18] Elaheh Maleki, Farouk Belkadi, Nikoletta Boli, Berend Jan Van Der Zwaag, Kosmas Alexopoulos, Spyridon Koukas, Mihai Marin-Perianu, Alain Bernard, and Dimitris Mourtzis. Ontology-based framework enabling smart product-service systems: Application of

sensing systems for machine health monitoring. IEEE Internet of Things Journal, 5(6):4496–4505, 2018.

[MBC+04] Andreas F Molisch, Kannan Balakrishnan, Chia-Chin Chong, Shahriar Emami, Andrew Fort, Johan Karedal, Juergen Kunisch, Hans Schantz, Ulrich Schuster, and Kai Siwiak. Ieee 802.15. 4a channel model-final report. IEEE P802, 15(04):0662, 2004.

[MBD+12] Andrew McAfee, Erik Brynjolfsson, Thomas H Davenport, DJ Patil, and Dominic Barton. Big data: the management revolution. Harvard business review, 90(10):60–68, 2012.

[McC15] TJ McCue. $117 billion market for internet of things in healthcare by 2020. Forbes Tech, April, 2015.

[MD03] Patrick Martin and Alain D'Acunto. Design of a production system: an application of integration product-process. International Journal of Computer Integrated Manufacturing, 16(7-8):509–516, 2003.

[MD08a] M-H Masson and T. Denœux. ECM: An evidential version of the fuzzy c-means algorithm. Pattern Recognition, 41(4):1384–1397, 2008.

[MD08b] Marie-Hélène Masson and Thierry Denoeux. Ecm: An evidential version of the fuzzy c-means algorithm. Pattern Recognition, 41(4):1384–1397, 2008.

[Mer10] David Mercier. Extending the contextual discounting of a belief function thanks to its canonical disjunctive decomposition. In 1st Workshop on Belief Functions, 2010.

[Mie12] Kaisa Miettinen. Nonlinear multiobjective optimization, volume 12. Springer Science & Business Media, 2012.

[MJ08] M McCann and A Johnston. Secom dataset, uci machine learning repository, 2008.

[MJ+10] Stefania Montani, Lakhmi C Jain, et al. Successful case-based reasoning applications, volume 305. Springer, 2010.

[MK10] Aristeidis Matsokis and Dimitris Kiritsis. An ontology-based approach for product lifecycle management. Computers in industry, 61(8):787–797, 2010.

[MQD08] David Mercier, Benjamin Quost, and Thierry Denœux. Refined modeling of sensor reliability in the belief function framework using contextual discounting. Information fusion, 9(2):246–258, 2008.

[MRAFH14] D Martín, Alejandro Rosete, Jesús Alcalá-Fdez, and Francisco Herrera. Qar-cip-nsga-ii: A new multi-objective evolutionary algorithm to mine quantitative association rules. Information Sciences, 258:1–28, 2014.

[MRH02] George Morcous, H Rivard, and AM Hanna. Modeling bridge deterioration using case-based reasoning. Journal of Infrastructure Systems, 8(3):86–95, 2002.

[MVH+04] Deborah L McGuinness, Frank Van Harmelen, et al. Owl web ontology language overview. W3C recommendation, 10(10):2004, 2004.

[Nad13] Yasser A Nada. A novel expert system for building house cost estimation: Design, implementation, and evaluation. constraints, 4(8), 2013.

[NB17] David Lira Nunez and Milton Borsato. An ontology-based model for prognostics and health management of machines. Journal of Industrial Information Integration, 6:33–46, 2017.

[NB18] David Lira Nuñez and Milton Borsato. Ontoprog: An ontology-based model for implementing prognostics health management in mechanical machines. Advanced Engineering Informatics, 38:746–759, 2018.

[NCF+03] Natalya Fridman Noy, Monica Crubézy, Ray W Fergerson, Holger Knublauch, Samson W Tu, Jennifer Vendetti, and Mark A Musen. Protégé-2000: an open-source ontology-development and knowledge-acquisition environment. In AMIA... Annual

Symposium proceedings. AMIA Symposium, volume 2003, pages 953–953. American Medical Informatics Association, 2003.

[NM+01] Natalya F Noy, Deborah L McGuinness, et al. Ontology development 101: A guide to creating your first ontology, 2001.

[OBGM11] Mohamed-Zied Ouertani, Salah Baïna, Lilia Gzara, and Gérard Morel. Traceability and management of dispersed product knowledge during design and manufacturing. Computer-Aided Design, 43(5):546–562, 2011.

[Obr03] Leo Obrst. Ontologies for semantically interoperable systems. In Proceedings of the twelfth international conference on Information and knowledge management, pages 366–369. ACM, 2003.

[OD10] Martin J O'Connor and Amar K Das. A method for representing and querying temporal information in owl. In International joint conference on biomedical engineering systems and technologies, pages 97–110. Springer, 2010.

[OFGC17] G Ortega, Ernestas Filatovas, EM Garzón, and Leocadio G Casado. Non-dominated sorting procedure for pareto dominance ranking on multicore cpu and/or gpu. Journal of Global Optimization, 69(3):607–627, 2017.

[OKTM05] Martin O'connor, Holger Knublauch, Samson Tu, and Mark Musen. Writing rules for the semantic web using swrl and jess. Protégé With Rules WS, Madrid, 2005.

[PB95] Nikhil R Pal and James C Bezdek. On cluster validity for the fuzzy c-means model. IEEE Transactions on Fuzzy systems, 3(3):370–379, 1995.

[PC03] Jd P Pavon and Sunghyun Choi. Link adaptation strategy for ieee 802.11 wlan via received signal strength measurement. In IEEE International Conference on Communications, 2003. ICC'03., volume 2, pages 1108–1113. IEEE, 2003.

[PC09] Panagiotis Papadopoulos and Liana Cipcigan. Wind turbines' condition monitoring: an ontology model. In 2009 International Conference on Sustainable Power Generation and Supply, pages 1–4. IEEE, 2009.

[PDS05] Lalit Patil, Debasish Dutta, and Ram Sriram. Ontology-based exchange of product data semantics. IEEE Transactions on automation science and engineering, 2(3):213–225, 2005.

[PDT12] Hervé Panetto, Michele Dassisti, and Angela Tursi. Onto-pdm: Product-driven ontology for product data management interoperability within manufacturing process environment. Advanced Engineering Informatics, 26(2):334–348, 2012.

[PDZ10] Ying Peng, Ming Dong, and Ming Jian Zuo. Current status of machine prognostics in condition-based maintenance: a review. The International Journal of Advanced Manufacturing Technology, 50(1-4):297–313, 2010.

[Pec09] Michael Pecht. Prognostics and health management of electronics. Encyclopedia of Structural Health Monitoring, 2009.

[Pec10] Michael G Pecht. A prognostics and health management roadmap for information and electronics-rich systems. IEICE ESS Fundamentals Review, 3(4):4_25–4_32, 2010.

[PHW02] Jian Pei, Jiawei Han, and Wei Wang. Mining sequential patterns with constraints in large databases. In Proceedings of the eleventh international conference on Information and knowledge management, pages 18–25, 2002.

[PLLL+08] Pedro Peris-Lopez, Tieyan Li, Tong-Lee Lim, Julio C Hernandez-Castro, and Juan M Estevez-Tapiador. Vulnerability analysis of a mutual authentication scheme under the epc class-1 generation-2 standard. In Workshop on RFID Security, volume 9, page 11, 2008.

[PMM+94] CB PERAKATH, CP MENZEL, RJ MAYER, et al. The idef5 ontology description capture method overview. Knowledge Based Systems, 1994.

[Pro11] Mark Proctor. Drools: a rule engine for complex event processing. In International Symposium on Applications of Graph Transformations with Industrial Relevance, pages 2–2. Springer, 2011.

[PVdBW+04] Davy Preuveneers, Jan Van den Bergh, Dennis Wagelaar, Andy Georges, Peter Rigole, Tim Clerckx, Yolande Berbers, Karin Coninx, Viviane Jonckers, and Koen De Bosschere. Towards an extensible context ontology for ambient intelligence. In European Symposium on Ambient Intelligence, pages 148–159. Springer, 2004.

[PVGPSF14] María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. Oops!(ontology pitfall scanner!): An on-line tool for ontology evaluation. International Journal on Semantic Web and Information Systems (IJSWIS), 10(2):7–34, 2014.

[Qui86] J. Ross Quinlan. Induction of decision trees. Machine learning, 1(1):81–106, 1986.

[R+01] Irina Rish et al. An empirical study of the naive bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence, volume 3, pages 41–46, 2001.

[Rao96] BKN Rao. Handbook of condition monitoring. Elsevier, 1996.

[Rey15] Douglas Reynolds. Gaussian mixture models. Encyclopedia of biometrics, pages 827–832, 2015.

[RGGMT+16] Sergio Ramírez-Gallego, Salvador García, Héctor Mouriño-Talín, David Martínez-Rego, Verónica Bolón-Canedo, Amparo Alonso-Betanzos, José Manuel Benítez, and Francisco Herrera. Data discretization: taxonomy and big data challenge. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 6(1):5–21, 2016.

[RHS05] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. WACV/MOTION, 2, 2005.

[RNB01] Michael J Roemer, EO Nwadiogbu, and G Bloor. Development of diagnostic and prognostic technologies for aerospace health management applications. In 2001 IEEE Aerospace Conference Proceedings (Cat. No. 01TH8542), volume 6, pages 3139–3147. IEEE, 2001.

[RP16] Petar Ristoski and Heiko Paulheim. Semantic web in data mining and knowledge discovery: A comprehensive survey. Web semantics: science, services and agents on the World Wide Web, 36:1–22, 2016.

[RPKC11] Catherine Roussey, Francois Pinet, Myoung Ah Kang, and Oscar Corcho. An introduction to ontologies and ontology engineering. In Ontologies in Urban development projects, pages 9–38. Springer, 2011.

[SB11] Zach Shelby and Carsten Bormann. 6LoWPAN: The wireless embedded Internet, volume 43. John Wiley & Sons, 2011.

[SCR+12] Abhinav Saxena, José R Celaya, Indranil Roychoudhury, Sankalita Saha, Bhaskar Saha, and Kai Goebel. Designing data-driven battery prognostic approaches for variable loading profiles: Some lessons learned. In European conference of prognostics and health management society, pages 72–732, 2012.

[SEA+02] York Sure, Michael Erdmann, Jürgen Angele, Steffen Staab, Rudi Studer, and Dirk Wenke. Ontoedit: Collaborative ontology development for the semantic web. In International Semantic Web Conference, pages 221–235. Springer, 2002.

[Sha76a] G. Shafer. A Mathematical Theory of Evidence. Princeton University Press, 1976.

[Sha76b] Glenn Shafer. A mathematical theory of evidence, volume 42. Princeton university press, 1976.

[SHM11] JZ Sikorska, Melinda Hodkiewicz, and Lin Ma. Prognostic modelling options for remaining useful life estimation by industry. Mechanical systems and signal processing, 25(5):1803–1836, 2011.

[Sik06] Marek Sikora. Rule quality measures in creation and reduction of data rule models. In International Conference on Rough Sets and Current Trends in Computing, pages 716–725. Springer, 2006.

[Sim99] Peter Simons. Parts: A study in ontology. 1999.

[SK94] P. Smets and R. Kennes. The Transferable Belief Model. Artificial Intelligence, 66(2):191–234, 1994.

[SL13] Thabet Slimani and Amor Lazzez. Sequential mining: patterns and algorithms analysis. arXiv preprint arXiv:1311.0350, 2013.

[SMASC09] Cesar Sanín, Leonardo Mancilla-Amaya, Edward Szczerbicki, and Paul CayfordHowell. Application of a multi-domain knowledge structure: The decisional dna. In Intelligent systems for knowledge management, pages 65–86. Springer, 2009.

[Sme05] P. Smets. Decision making in the TBM : The necessity of the pignistic transformation. International Journal of Approximate Reasoning, 38:133–147, 2005.

[SMH+15] Rainer Schmidt, Michael Möhring, Ralf-Christian Härting, Christopher Reichstein, Pascal Neumaier, and Philip Jozinović. Industry 4.0-potentials for creating smart products: empirical research results. In International Conference on Business Information Systems, pages 16–27. Springer, 2015.

[SMMS02] Ljiljana Stojanovic, Alexander Maedche, Boris Motik, and Nenad Stojanovic. User-driven ontology evolution management. In International Conference on Knowledge Engineering and Knowledge Management, pages 285–300. Springer, 2002.

[SMS+19] Chayma Sellami, Carlos Miranda, Ahmed Samet, Mohamed Anis Bach Tobji, and François de Beuvron. On mining frequent chronicles for machine failure prediction. Journal of Intelligent Manufacturing, pages 1–17, 2019.

[Smu95] Raymond M Smullyan. First-order logic. Courier Corporation, 1995.

[Smu12] Raymond R Smullyan. First-order logic, volume 43. Springer Science & Business Media, 2012.

[SS01] Bernhard Scholkopf and Alexander J Smola. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press, 2001.

[SS08] Florin Stoica and Dana Simian. Approaches to cognitive support in biomedical knowledge-based systems. In Proceedings of the 9th WSEAS International Conference on Mathematics & Computers In Biology & Chemistry, pages 217–222. World Scientific and Engineering Academy and Society (WSEAS), 2008.

[SS09] Cesar Sanin and Edward Szczerbicki. Experience-based knowledge representation: Soeks. Cybernetics and Systems: an international journal, 40(2):99–122, 2009.

[SSA+00] A Th Schreiber, Guus Schreiber, Hans Akkermans, Anjo Anjewierden, Nigel Shadbolt, Robert de Hoog, Walter Van de Velde, Nigel R Shadbolt, and Bob Wielinga. Knowledge engineering and management: the CommonKADS methodology. MIT press, 2000.

[SSGR09] Paulo Sampaio, Pedro Saraiva, and António Guimarães Rodrigues. Iso 9001 certification research: questions, answers and approaches. International Journal of Quality & Reliability Management, 26(1):38–58, 2009.

[SSP+14] Gian Antonio Susto, Andrea Schirru, Simone Pampuri, Seán McLoone, and Alessandro Beghi. Machine learning for predictive maintenance: A multiple classifier approach. IEEE Transactions on Industrial Informatics, 11(3):812–820, 2014.

[SSS82] Motoi Suwa, A Carlisle Scott, and Edward H Shortliffe. An approach to verifying completeness and consistency in a rule-based expert system. Ai Magazine, 3(4):16–16, 1982.

[SST+00] Craig Schlenoff, Craig Schlenoff, Florence Tissot, John Valois, and Jintae Lee. The process specification language (PSL) overview and version 1.0 specification. Citeseer, 2000.

[SST18] Chayma Sellami, Ahmed Samet, and Mohamed Anis Bach Tobji. Frequent chronicle mining: Application on predictive maintenance. In 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pages 1388–1393. IEEE, 2018.

[Ste85] Luc Steels. Second generation expert systems. Future generation computer systems, 1(4):213–221, 1985.

[Ste16] Renee Stevens. Engineering mega-systems: The challenge of systems engineering in the information age. Auerbach Publications, 2016.

[Sto74] Mervyn Stone. Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society: Series B (Methodological), 36(2):111–133, 1974.

[STV+07] C Sanin, C Toro, J Vaquero, E Szczerbicki, and J Posada. Implementing decisional dna in industrial maintenance by a knowledge soupa extension. Systems Science, 33(2):61–68, 2007.

[SW11] Marek Sikora and Łukasz Wróbel. Data-driven adaptive selection of rules quality measures for improving the rules induction algorithm. In International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing, pages 278–285. Springer, 2011.

[SWdH+94] Guus Schreiber, Bob Wielinga, Robert de Hoog, Hans Akkermans, and Walter Van de Velde. Commonkads: A comprehensive methodology for kbs development. IEEE expert, 9(6):28–37, 1994.

[SWHZ11] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation–a review on the statistical data driven approaches. European journal of operational research, 213(1):1–14, 2011.

[SWW15] Ahmad-Reza Sadeghi, Christian Wachsmann, and Michael Waidner. Security and privacy challenges in industrial internet of things. In 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), pages 1–6. IEEE, 2015.

[SWWP10] J Schwarzenbach, L Wilkinson, M West, and M Pilling. Mapping the remote condition monitoring architecture. Research Programme. Rail Safety and Standards Boards (RSSB) LTD. RSSB Core Report, 2010.

[TDS03] David Tranfield, David Denyer, and Palminder Smart. Towards a methodology for developing evidence-informed management knowledge by means of systematic review. British journal of management, 14(3):207–222, 2003.

[TGP+09] Carlos Toro, Manuel Graña, Jorge Posada, Cesar Sanín, and Edward Szczerbicki. An architecture for the semantic enhancement of virtual engineering applications. In Intelligent Systems for Knowledge Management, pages 175–195. Springer, 2009.

[TS07] Patrick P Tsang and Sean W Smith. A low-latency, high-integrity security retrofit for legacy scada systems. Technical report, Technical Report TR2007-603, Department of Computer Science, Dartmouth College, 2007.

[TSC+12] Carlos Toro, Eider Sanchez, Eduardo Carrasco, Leonardo Mancilla-Amaya, Cesar Sanín, Edward Szczerbicki, Manuel Graña, Patricia Bonachela, Carlos Parra, Gloria Bueno, et al. Using set of experience knowledge structure to extend a rule set of clinical decision support system for alzheimer's disease diagnosis. Cybernetics and Systems, 43(2):81–95, 2012.

[UGL08] Serdar Uckun, Kai Goebel, and Peter JF Lucas. Standardizing research methods for prognostics. In 2008 International Conference on Prognostics and Health Management, pages 1–10. IEEE, 2008.

[UK95] Michael Uschold and Martin King. Towards a methodology for building ontologies. Citeseer, 1995.

[UYC+11] Zahid Usman, Robert Ian Marr Young, Nitishal Chungoora, Claire Palmer, Keith Case, and Jenny Harding. A manufacturing core concepts ontology for product lifecycle interoperability. In International IFIP Working Conference on Enterprise Interoperability, pages 5–18. Springer, 2011.

[UYC+13] Zahid Usman, RIM Young, Nitishal Chungoora, Claire Palmer, Keith Case, and Jennifer A Harding. Towards a formal manufacturing reference ontology. International Journal of Production Research, 51(22):6553–6572, 2013.

[VHL05] Marcela Vegetti, Gabriela P Henning, and Horacio P Leone. Product ontology: definition of an ontology for the complex product modelling domain. In Proceedings of the Mercosur Congress on Process Systems Engineering, 2005.

[Wan16] K Wang. Intelligent predictive maintenance (ipdm) system–industry 4.0 scenario. WIT Transactions on Engineering Sciences, 113:259–268, 2016.

[WBG08] Karen Walzer, Tino Breddin, and Matthias Groch. Relative temporal constraints in the rete algorithm for complex event detection. In Proceedings of the second international conference on Distributed event-based systems, pages 147–155. ACM, 2008.

[WKM04] Andrew K Wright, John A Kinast, and Joe McCarty. Low-latency cryptographic protection for scada communications. In International Conference on Applied Cryptography and Network Security, pages 263–277. Springer, 2004.

[WSJ17] Martin Wollschlaeger, Thilo Sauter, and Juergen Jasperneite. The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0. IEEE industrial electronics magazine, 11(1):17–27, 2017.

[WXY+17] Zhenyu Wu, Yuan Xu, Yunong Yang, Chunhong Zhang, Xinning Zhu, and Yang Ji. Towards a semantic web of things: a hybrid semantic annotation, extraction, and reasoning framework for cyber-physical system. Sensors, 17(2):403, 2017.

[WY07] Achmad Widodo and Bo-Suk Yang. Support vector machine in machine condition monitoring and fault diagnosis. Mechanical systems and signal processing, 21(6):2560–2574, 2007.

[WZG+04] Xiaohang Wang, Daqing Zhang, Tao Gu, Hung Keng Pung, et al. Ontology based context modeling and reasoning using owl. In Percom workshops, volume 18, page 22. Citeseer, 2004.

[XP09] Yang Xiao and Yi Pan. Emerging wireless LANs, wireless PANs, and wireless MANs: IEEE 802.11, IEEE 802.15, 802.16 wireless standard family, volume 57. John Wiley & Sons, 2009.

[XYWV12] Feng Xia, Laurence T Yang, Lizhe Wang, and Alexey Vinel. Internet of things. International Journal of Communication Systems, 25(9):1101, 2012.

[YISI00] Mariko Yoshida, Tetsuya Iizuka, Hisako Shiohara, and Masanori Ishiguro. Mining sequential patterns including time intervals. In Data Mining and Knowledge Discovery: Theory, Tools, and Technology II, volume 4057, pages 213–220. International Society for Optics and Photonics, 2000.

[ZG09] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning, 3(1):1–130, 2009.

[ZLZ15] Keliang Zhou, Taigang Liu, and Lifeng Zhou. Industry 4.0: Towards future industrial opportunities and challenges. In 2015 12th International conference on fuzzy systems and knowledge discovery (FSKD), pages 2147–2152. IEEE, 2015.

[ZM15] Cecilia Zanni-Merk. Krem: A generic knowledge-based framework for problem solving in engineering. In KEOD, pages 381–388, 2015.

[ZMMZWdB15] Cecilia Zanni-Merk, Stella Marc-Zwecker, Cédric Wemmert, and François de Bertrand de Beuvron. A layered architecture for a fuzzy semantic approach for satellite image analysis. International Journal of Knowledge and Systems Science (IJKSS), 6(2):31–56, 2015.

[ZMS19] Cecilia Zanni-Merk and Edward Szczerbicki. Building collective intelligence through experience: a survey on the use of the krem

model. Journal of Intelligent & Fuzzy Systems, (Preprint):1–13, 2019.

[ZQT+15] Yin Zhang, Meikang Qiu, Chun-Wei Tsai, Mohammad Mehedi Hassan, and Atif Alamri. Health-cps: Healthcare cyber-physical system assisted by cloud and big data. IEEE Systems Journal, 11(1):88–95, 2015.

[ZSS10] Haoxi Zhang, Cesar Sanin, and Edward Szczerbicki. Decisional dna-based embedded systems: A new perspective. Systems Science, 36(1):21–26, 2010.

[ZSS12] Haoxi Zhang, Cesar Sanín, and Edward Szczerbicki. Making digital tv smarter: capturing and reusing experience in digital tv. Cybernetics and Systems, 43(2):127–135, 2012.

[ZYZ15] Anmei Zhou, Dejie Yu, and Wenyi Zhang. A research on intelligent fault diagnosis of wind turbines based on ontology and fmeca. Advanced Engineering Informatics, 29(1):115–125, 2015.