Towards Autonomic and Cognitive IoT Systems, Application to Patients' Treatments Management

Emna Mezghani

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)*

Présentée et soutenue le *15/12/2016* **par :**

Emna MEZGHANI

Towards Autonomic and Cognitive IoT Systems, Application to Patients' Treatments Management

JURY

Miriam CAPRETZ	Université de Western Ontario	Invitée
Myriam LAMOLLE	Université Paris 8	Rapporteur
Michael MRISSA	Université de Pau	Rapporteur
Mohamed MOSBAH	Bordeaux INP	Examinateur
Walid GAALOUL	Télécom SudParis	Examinateur
Marcos DA SILVEIRA	LIST	Examinateur
Cédric PRUSKI	LIST	Examinateur
Ernesto EXPOSITO	Université de Pau	Codirecteur de thèse
Khalil DRIRA	LAAS-CNRS	Directeur de thèse

École doctorale et spécialité :
MITT : Domaine STIC : Réseaux, Télécoms, Systèmes et Architecture
Unité de Recherche :
LAAS - CNRS (UPR 8001)
Luxembourg Institute of Science and Technology (LIST)
Directeur(s) de Thèse :
Khalil DRIRA et *Ernesto EXPOSITO*
Rapporteurs :
Myriam LAMOLLE et *Michael MRISSA*

Acknowledgments

First, I would like to thank the "*Luxembourg National Research Fund (FNR)*" for funding my PhD thesis. Without such a grant this work would not have been realized.

I would like to express my sincere gratitude to my thesis director Dr. Khalil Drira for his support, encouragement, and invaluable guidance. I am deeply thankful for the opportunity that you offered me to work with you, and to learn a lot from you. Your vision, thoughts and confidence have inspired me to go further the challenges I have encountered, and taught me how to grow as a researcher.

I feel tremendously lucky to have Prof. Ernesto Exposito as my thesis co-director. Our brainstorming and continuous discussions sharply increased the productivity of my research, and helped me developing new skills. I truly appreciate all the support and the freedom you gave me to shape my research work.

I would like also to thank Dr. Marcos Da Silveira and Dr. Cédric Pruski for their support and help during my stay at Luxembourg, and for the time they spent with me to discuss and to provide me valuable suggestions.

My deep gratitude is also addressed to the members of jury: Prof. Myriam Lamolle and Prof. Michael Mrissa for reviewing my dissertation and providing valuable comments; Prof. Mohamed Mosbah for presiding the jury; and Prof. Walid Gaaloul and Prof. Miriam Capretz for accepting being part of my PhD committee members.

I particularly want to thank all the SARA team members at LAAS-CNRS for the moments that we spent together and exchanged ideas. Special thanks go to Ghada, Mahdi, Slim, Aymen, Mohamed, Mohamed Amin, Nouha, Codé, Jorge, Yassine, Tom, Riadh, Ismail and Emna. I would like to thank also my colleagues at the LIST. Special thanks go to Wided for her precious advices and recommendations, Moussa for his support, Abir, Silvio and Uwe for the enjoyable discussions we have made during my stay at LIST. I would like to extend my thanks to Dr. Katarina Grolinger for the fruitful collaboration and discussions that we have made. Special thanks go also to Doctor Mouna Elleuch, endocrinologist at the *Hedi Chaker Hospital* in Sfax-Tunisia, for her collaboration.

It is with great emotion that I dedicate the entire work to my parents and my brothers. You are my source of strength that keeps me up. Without your advices and support, I would never reach this grade and get the diploma.

Abstract

Nowadays, the adoption of the Internet of Things (IoT) drastically witnesses an increase in different domains, and contributes to the fast digitalization of the universe. Henceforth, next generation of IoT-based systems are set to become more complex to design and manage. Collecting real-time IoT generated data unleashes a new wave of opportunities for business to take more precise and accurate decisions at the right time. Nonetheless, a set of challenges including the complexity of IoT-based systems and the management of the ensuing big and heterogeneous data as well as the system scalability; need to be addressed for the development of flexible smart IoT-based systems that drive the business decision-making.

With respect to challenge which relates to the complexity of IoT management, we propose to automate the management of IoT-based systems based on an autonomic computing approach. However, autonomic computing alone is not enough for the development of smart IoT-based systems. Indeed, these systems should implement cognitive capabilities that allow them learning and generating decisions at the right time. Consequently, we propose a model-driven methodology for designing smart IoT-based systems. We defined within this methodology a set of *autonomic cognitive design patterns* that aim at (1) delineating the dynamic coordination of the management processes to deal with the system's context changeability and requirements evolution at run-time, and (2) adding cognitive abilities to IoT-based systems to understand big data and interact with human through generating new insights. Our ultimate goal was to assist the architect when designing flexible smart IoT-based systems by selecting the right pattern or combination of patterns to solve complex requirements.

With respect to challenges which relate to big data and scalability management, we propose a generic semantic big data platform that integrates heterogeneous distributed data sources deployed on the cloud, and generates knowledge that will be exposed as a service (*Knowledge as a Service–KaaS*). The proposed architecture represents an extension of the NIST Big Data and Cloud Computing reference architectures with a semantic layer that enables the machines collecting and interpreting the received data, curating and harmonizing it for better analytic and visualization. More specifically, we are interested in healthcare as an applicative domain. Thus, based on big data tools for data stream processing, we proposed a cognitive monitoring system implementing a combination of the proposed patterns for managing the patient health based on wearables and promptly detecting personalized anomalies. Hence, we elaborated the *Wearable Healthcare Ontology (WH_O)* for the integration of heterogeneous wearable data. The proposed system is deployed within the KaaS, and its performance (in terms of response time and scalability) when processing huge amount of heterogeneous data streams has been evaluated following different KaaS configurations.

Finally, to provide smart IoT-based systems able to reason and generate recommendations, we enriched the proposed system with new cognitive mechanisms including the medical procedural knowledge and the personalization process. Thus, a methodology for extracting and formalizing the medical knowledge based on the collaboration of medical experts is proposed. The output of this methodology is a flexible semantic model named *Treatment Plan Ontology (TPO)* that describes the medical interventions. We also defined an *Ontology-based planning algorithm* that integrates TPO with external existing knowledge sources in order to provide personalized decisions concerning the patient treatment. We evaluated the proposed algorithm through simulating real clinical use cases and comparing the generated recommendations to the experts' advice. We highlighted also the system performance on the cloud, and provided recommendations for selecting the appropriate IT configuration based on the system requirements.

List of publications

International Journals

- Emna Mezghani, Ernesto Exposito, Khalil Drira, A collaborative methodology for tacit knowledge management: Application to scientific research, Future Generation Computer Systems, Available online 16 May 2015, ISSN 0167-739X, doi: http://dx.doi.org/10.1016/j.future.2015.05.007

- Emna Mezghani, Ernesto Exposito, Khalil Drira, Marcos Da Silveira, Cédric Pruski. A Semantic Big Data Platform for Integrating Heterogeneous Wearable Data in Healthcare, Journal of Medical Systems, 2015, doi: 10.1007/s10916-015-0344-x

- Katarina Grolinger, Emna Mezghani, Miriam Capretz and Ernesto Exposito. Collaborative knowledge as a service applied to the disaster management domain, International Journal of Cloud Computing, 2013, doi: 10.1504/IJCC.2015.067706

Book Chapters

- Katarina Grolinger, Emna Mezghani, Miriam Capretz and Ernesto Exposito: Knowledge as a Service Framework for Collaborative Data Management in Cloud Environments - Disaster Domain, in Book: Managing Big Data in Cloud Computing Environments, IGI Global 2016, doi: 10.4018/978-1-4666-9834-5.ch008

- Emna Mezghani, Riadh Ben Halima, Khalil Drira: DRAAS: Dynamically Reconfigurable Architecture for Autonomic Services. In Book: Web Services Foundations 2014, Springer,ISBN 978-1-4614-7517-0, pp.483-505, doi: 10.1007/978-1-4614-7518-7_19

International Conferences

- Emna Mezghani, Marcos Da Silveira, Cédric Pruski, Ernesto Exposito and Khalil Drira: An Ontology-driven Adaptive System for the Patient Treatment Management, 28th International Conference on Software Engineering & Knowledge Engineering (SEKE), San Francisco Bay, California, USA, July 1 - July 3, 2016, doi: 10.18293/SEKE2016-155

- Emna Mezghani: A Semantic Cloud-based Knowledge Mediator for Smart Clinical Decision Support System, CAL 2015, Revue des Nouvelles Technologies de l'Information, vol. RNTI-L-8, pp.47-60.

- Emna Mezghani, Marcos Da Silveira, Cédric Pruski, Ernesto Exposito, Khalil Drira: A perspective of adaptation in healthcare, 25th European Medical Informatics Conference (MIE), August 31-September 3, 2014 Istanbul, Turkey, 2014, pp. 206-210. doi: http://dx.doi.org/10.3233/978-1-61499-432-9-206

- Codé Diop, Aymen Kamoun, Emna Mezghani, Mohamed Zouari Ernesto Exposito. A smart Mediator to Integrate Dynamic Networked Enterprises. 7th International Conference on Interoperability Enterprises Systems and Applications, I-ESA 2014, March, 24-28 2014, doi: 10.1002/9781119081418.ch18

- Katarina Grolinger, Miriam Capretz, Emna Mezghani and Ernesto Exposito, Knowledge as a Service Framework for Disaster Data Management, Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2013 IEEE 22nd International Workshop on, Hammamet, 2013, pp. 313-318. doi: 10.1109/WETICE.2013.48

- Codé Diop, Emna Mezghani, Ernesto Exposito, Christophe Chassot, Khalil Drira: QoS-driven Autonomic Abilities through a Multi-homed Transport Protocol, Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on, Barcelona, 2013, pp. 645-652. doi: 10.1109/AINA.2013.104

- Emna Mezghani, Riadh Ben Halima, Ismael Bouassida Rodriguez, and Khalil Drira. 2013. A model driven methodology for enabling autonomic reconfiguration of service oriented architecture. InProceedings of the 28th Annual ACM Symposium on Applied Computing (SAC '13). ACM, New York, NY, USA, 1772-1773. doi: http://dx.doi.org/10.1145/2480362.2480695

- Emna Mezghani and Riadh Ben Halima, "DRF4SOA: A Dynamic Reconfigurable Framework for Designing Autonomic Application Based on SOA," Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2012 IEEE 21st International Workshop on, Toulouse, 2012, pp. 95-97. doi: 10.1109/WETICE.2012.81

Contents

Contents vii

List of Figures xi

List of Tables xiii

1 General Introduction 1
- 1.1 Introduction . . . 1
- 1.2 Healthcare Context . . . 3
- 1.3 Research Problems . . . 4
  - 1.3.1 Smart IoT Design Complexity . . . 4
  - 1.3.2 Knowledge and Big Data Challenges . . . 5
  - 1.3.3 System Scalability . . . 7
- 1.4 Existing Work . . . 8
- 1.5 Thesis Positioning . . . 9
- 1.6 Scientific Contributions . . . 10
- 1.7 Dissertation Outline . . . 12

2 Background & State of the Art 15
- 2.1 Introduction . . . 15
- 2.2 Model-driven Methodologies . . . 16
- 2.3 Autonomic System Design . . . 17
  - 2.3.1 The Autonomic Computing Vision . . . 17
  - 2.3.2 Patterns for the Design of Adaptive and Autonomic Systems 20
  - 2.3.3 Discussion . . . 27
- 2.4 Cognitive Computing Concepts . . . 28
  - 2.4.1 Knowledge for Cognitive Computing . . . 28
  - 2.4.2 Big Data Management . . . 34
  - 2.4.3 Discussion . . . 37
- 2.5 Cloud Computing . . . 38
  - 2.5.1 NIST Cloud Computing . . . 38
  - 2.5.2 Knowledge as a Service . . . 39
  - 2.5.3 Discussion . . . 40
- 2.6 Conclusion . . . 40

3 Autonomic Cognitive Design Patterns for the Design of Smart IoT-based Systems 41
3.1 Introduction . . . 42
3.2 IoT Healthcare System for Patient Treatment Management . . . 42
3.2.1 Autonomic Computing for Patient Treatment Management 43
3.2.2 Maturity Levels for Autonomic and Cognitive IoT-based Systems . . . 44
3.3 Design Patterns Identification . . . 46
3.3.1 Management Processes Level . . . 46
3.3.2 Data Management Level . . . 47
3.3.3 Infrastructure Management Level . . . 50
3.4 A Model-driven Methodology for the Design of Autonomic and Cognitive IoT-based Systems . . . 50
3.4.1 An Overview of the Proposed Methodology . . . 50
3.4.2 Management Processes Coordination Patterns . . . 52
3.4.3 Semantic Knowledge Mediator Pattern . . . 71
3.4.4 Big Data & Scalability Management Patterns . . . 73
3.5 Conclusion . . . 79

4 A Knowledge as a Service Platform for Heterogeneous Wearable Data Integration 81
4.1 Introduction . . . 82
4.2 Wearable Computing in Healthcare . . . 82
4.3 Existing IoT Platforms . . . 84
4.4 Knowledge as a Service for Semantic Cloud-based Big Data Management . . . 88
4.4.1 Overview of the KaaS Architecture . . . 88
4.4.2 Wearable Healthcare Ontology . . . 90
4.4.3 A Cognitive Monitoring System for Patient Health Management . . . 93
4.5 Performance Evaluation . . . 97
4.5.1 On-premises vs Cloud Evaluation . . . 97
4.5.2 Multi-Node Cloud Evaluation . . . 102
4.6 Conclusion . . . 106

5 A Prescriptive Cognitive System for Patient Treatment Management 109
5.1 Introduction . . . 110
5.2 Decision-making for Treatment Management . . . 110
5.2.1 Clinical Decision Support System . . . 111
5.2.2 Treatment Adaptation & Personalization . . . 111
5.2.3 Linked Data in Healthcare . . . 114

5.2.4 Discussion 115
5.3 A Prescriptive Cognitive System for Personalization 116
5.3.1 System Architecture Overview 116
5.3.2 A Collaborative Methodology for Medical Knowledge Formalization 117
5.3.3 Treatment Plan Ontology 120
5.3.4 A Collaborative Semantic Web Platform for Medical Knowledge Acquisition 123
5.3.5 Ontology-based Planning Algorithm for Treatment Personalization 126
5.4 Clinical Evaluation 129
5.4.1 Use Case 1 129
5.4.2 Use Case 2 130
5.4.3 Use Case 3 from Clinical Diabetes Journal 131
5.5 Performance Evaluation 133
5.6 Conclusion 137

6 Conclusion and Perspectives **139**
6.1 Conclusion 139
6.2 Perspectives 143

A Appendix **145**

References **147**

List of Figures

1.1 Next generation of healthcare system: patient-centric approach . . 4
1.2 Autonomic and cognitive IoT-based systems' challenges and approaches 10
1.3 Thesis main research areas 11
1.4 Dissertation structure 14

2.1 The autonomic computing reference architecture 19
2.2 DIKW pyramid 29
2.3 Main activities of knowledge management 30
2.4 The KaaS paradigm 39

3.1 Autonomic computing for managing patient treatment 43
3.2 Autonomic cognitive IoT-based system maturity levels 45
3.3 A model-driven methodology for the design of autonomic and cognitive IoT-based systems 51
3.4 Knowledge pattern 54
3.5 Instantiation of the knowledge pattern in healthcare 55
3.6 Cognitive monitoring management pattern for IoT and human interaction 56
3.7 Instantiation of the cognitive monitoring pattern in healthcare . . . 57
3.8 Predictive cognitive management pattern 59
3.9 Instantiation of predictive pattern in healthcare 60
3.10 Prescriptive cognitive management pattern 62
3.11 Instantiation of the Prescriptive Cognitive Management Pattern in Healthcare 63
3.12 Autonomic cognitive management pattern 65
3.13 Management Process Ontology (MPO) 65
3.14 Instantiation of the autonomic cognitive management pattern in healthcare 69
3.15 Semantic knowledge mediator pattern 72
3.16 Big data stream detection pattern 74
3.17 Big data analytic predictive pattern 76
3.18 Multi-tenant management process pattern (UML deployment diagram) 78

4.1 IoT Platforms' approaches 86

4.2 The generic conceptual KaaS architecture 89
4.3 Depiction of the main classes and relationships of the Wearable Healthcare Ontology 92
4.4 An expert of the collaborative semantic web platform for providers' characteristics annotation 94
4.5 Big data stream detection sequence diagram within the KaaS . . . 95
4.6 On-premises deployment of the cognitive monitoring system . . . 98
4.7 Performance evaluation on-premises deployment 99
4.8 Cognitive monitoring system deployment in cloud-config1 100
4.9 Comparison of the system performance on-premises vs cloud deployment 101
4.10 A multi-node deployment on the cloud 103
4.11 Performance evaluation of the cognitive monitoring system in production mode 104
4.12 CPU consumption in the Storm supervisors when processing 1,200,000 streams 105
4.13 Evaluation of the impact of IT resources configuration on system performance 106

5.1 A three-phase methodology for the Treatment Plan Ontology design and implementation 119
5.2 Depiction of the main classes and relationships with their cardinalities in the proposed Treatment Plan Ontology 122
5.3 The main component of the proposed collaborative semantic web platform 123
5.4 Semantic MediaWiki for medical intervention annotation 124
5.5 An expert of TPO instantiation describing diabetes treatments . . 125
5.6 The workflow for personalizing treatment decision 127
5.7 The ordered personalized treatments for use case 1 130
5.8 The ordered personalized treatments for use case 2 131
5.9 The ordered personalized treatments for use case 3 132
5.10 Evaluation of the planning response time per use case 134
5.11 The impact of integrating DrugBank on system scalability & response time 135
5.12 Impact of the CPU configuration on the planning performance . . 136

A.1 MPO visualization using VOWL 145
A.2 WH_O visualization using VOWL 146
A.3 TPO visualization using VOWL 146

List of Tables

2.1 A survey of works dealing with design patterns for autonomic and self-adaptive systems 26

3.1 Inventory of reusable patterns 49

4.1 A survey of IoT platforms: approaches and techniques 87
4.2 Performance measurements on-premises vs cloud deployment . . 101
4.3 Performance evaluation of the parallelism on the cloud 103
4.4 The cognitive monitoring system performance measurements on two cloud infrastructures 105

5.1 A survey on treatment adaptation and personalization 113
5.2 Planning performance evaluation based on the allocated IT resources 136

CHAPTER 1

General Introduction

"If you can't explain it simply, you don't understand it well enough."

- Albert Einstein

Contents

1.1 Introduction 1
1.2 Healthcare Context 3
1.3 Research Problems 4
1.3.1 Smart IoT Design Complexity 4
1.3.2 Knowledge and Big Data Challenges 5
1.3.3 System Scalability 7
1.4 Existing Work 8
1.5 Thesis Positioning 9
1.6 Scientific Contributions 10
1.7 Dissertation Outline 12

1.1 Introduction

The human brain is the quintessential complex system in the world. Its amazing energy and supernatural abilities impelled many research activities in cognitive science including psychology, neuroscience and philosophy [1, 2, 3] to study and model its activities and structure to better understand how it instantly perceives the external world, processes and interprets the massively complex received data, and aggregates it with already stored information to deduce the action to do. Computing systems have been influenced by these potential findings and proposed new approaches that mimic the human brain to realize leap forward smart systems. They have been evolved from imperative computing, to autonomic computing, to

reach cognitive computing [4]. While imperative computing includes traditional programming with passive technologies, autonomic computing [5] introduces self-management mechanisms that automatically adapt the system behaviours based on its context changes. Freshly, cognitive computing has been introduced for the development of intelligent systems able to perceive, learn and think as human do [6]. Perhaps, IBM Watson [7] is the most famous question answering cognitive system that won the Jeopardy in February 2011.

Industrially speaking, computing systems have made great strides, and are gaining further recognition with the integration of a new phenomenon called the Internet of Things (IoT) where devices including sensors and actuators are sharing the observations and communicating through the internet. Nowadays, the IoT market witnesses an important increase. According to Gartner [8], around 25 Billion Connected "Things" will be in use in 2020. The integration of IoT is popular in different domains such as healthcare, smart cities, autonomous cars, etc. For instance, Apple proposed the HealthKit[1] that allows developing healthcare[2] apps for continuously monitoring the patient health based on sensors integrated in the Apple watch.

Analogously to human sensors, these novel technologies churn out myriad of data through enabling objects sensing the physical world. The effective use of IoT is much more than just connecting things, it encompasses as a primary concern the management and the transformation of the IoT generated data into insights and business benefits [9]. Recently, a new era of IoT called "Cognitive IoT"[10] has been enunciated. It aims at integrating cognitive technologies into IoT-based systems to ensure the smart management through enabling the cooperation and interaction between IoT and human. So that IoT-based systems may learn from human intelligence and provide more accurate analytic.

Clearly, attaining IoT values lean on how the IoT data is analyzed and integrated with external sources in order to unleash a new wave of opportunities for business and people to take real-time and accurate decisions. However, IoT proliferation poses significant challenges pertaining to the complexity of designing and managing such systems, to the heterogeneity of the generated data, to the scalability and the flexibility of the system to support the integration of highly distributed and heterogeneous data and knowledge sources.

The ultimate goals of this chapter are:

- Introducing the role of IoT in healthcare, and their importance in delivering patient-centric management (section 1.2).
- Highlighting the main challenges that should be considered to conceive smart IoT-based systems able to understand the collected data, and take the right

[1] https://developer.apple.com/healthkit/

[2] http://www-05.ibm.com/innovation/uk/watson/watson_in_healthcare.shtml

decision at the right time based on the domain knowledge (section 1.3).

- Presenting our position compared to existing works as well as the main approaches that cover the identified challenges (section 1.4 and section 1.5).
- Finally introducing our thesis' contributions and presenting the dissertation structure (section 1.6 and section 1.7).

1.2 Healthcare Context

Advances in healthcare technologies have transformed the patient care management from paper-based into computer-based through the implementation of Electronic Health Records (EHRs) [11]. EHRs store the patient medical and treatment histories in a digital format, which make the health information available, sharable and up-to-date for better decisions. Additionally, with the development of the internet and web technologies, medical knowledge becomes available, easy to share and reuse. It describes new discovered treatments, drugs characteristics and disease management strategies in order to drive decision-making. For instance, DrugBank [12] is an example of accessible medical knowledge describing drug-drug and drug-food interactions which help personalizing the patient treatment.

Recently, the emergency of IoT has revolutionized healthcare through capturing real-time and individualized data concerning patients. It fosters providing patient centric management as well as preventing health complications. The wearable computing, as an example of IoT technologies, is fast gaining momentum. A recent study conducted by ABI Research[3] estimates that the global market for wearable devices in health and fitness could reach 169.5 million devices by 2017. The use of wearable computing allows enhancing the patient's care and life style management through enabling remote data stream processing and detecting anomalies at the right time which profoundly impacts the decision process. Its utility goes further to enable disabled person's feeling motions and doing actions, despite their impairment [13].

Real-time analyzing the data stemming from the patient wearable devices, and integrating them with the patient medical history and with medical knowledge could far-reaching benefits for accelerating the decision-making and personalizing the patient treatment. Figure 1.1 illustrates the next generation of smart IoT-based healthcare system for the patient centric management. It portrays the main actors, as well as the data and knowledge sources that should be integrated to ensure such purpose. However, a set of challenges, which will be detailed in the next section, need to be overcome to guarantee the integration and provide better quality of care to patients.

[3] https://www.abiresearch.com/press/wearable-sports-and-fitness-devices-will-hit-90-mi/

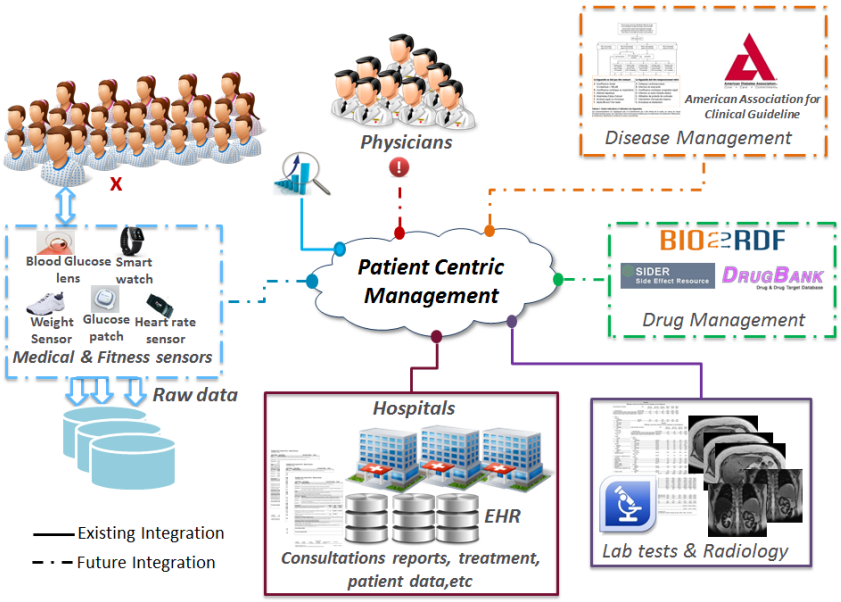

Figure 1.1: Next generation of healthcare system: patient-centric approach

1.3 Research Problems

The proliferation of IoT has given rise to new challenges pertaining to the complexity of designing and managing the dynamic evolution of systems, to the big data and to the system scalability.

1.3.1 Smart IoT Design Complexity

The wide adoption of IoT yields to more complex and heterogeneous systems of systems (SoS) [14] that should interact with each other to meet the system requirements. As these technologies sense the physical world, it is easy to detect context changes but hard to manage. Thus, automating the management of IoT-based systems may alleviate the system complexity, accelerate and facilitate interactions with domain experts for better decision making. Hence, two main challenges need to be addressed when conceiving smart IoT-based systems:

Context changeability: Complex systems implementing real-world applications continuously evolve and generate unforeseen requirements [15]. For instance, in healthcare, we consider managing patients with diabetes as a primary requirement. During treating diabetes, because of aging and biological changes, a new requirement such as managing the hypertension may occur. Thus, new processes for controlling hypertension need to be integrated into the system. However, traditional management systems are considered

as ad-hoc systems –designed and implemented from scratch, which impedes the dynamic self-management and requires additional human efforts. Smart self-managed systems should be able to support the dynamic discovery of the processes, their composition and coordination to manage more complex situations, especially those unpredictable at design-time.

System flexibility: Integrating new sensors at run-time to monitor and collect more data about the system may require re-engineering the system for integration purposes due to the heterogeneity of the data. This yields to a high design and implementation cost. Thus, the evolution of the system architecture should be considered at design-time in order to provide more flexible IoT-based systems. Consequently, conceptual models and patterns, which elucidate how to manage IoT-based system evolution, are required to effectively assist software architect providing efficient solutions based on the system's requirements.

1.3.2 Knowledge and Big Data Challenges

Despite the ability to detect the context changes, a smart IoT-based system should be able to take the right decision at the right time in order to assist the domain experts with business benefits. Thus, they should interact with human to learn from their deep knowledge of the domain and to provide hidden knowledge about the system from the gathered data.

1.3.2.1 Knowledge Challenges

Knowledge is the key element for an efficient decision-making. It is located in documents, videos, images and in human mind. With the development of the Web 2.0, domain knowledge is being published in the web for sharing and reuse by human. Nowadays, large-scale knowledge bases are available, but with unstructured format, which make manually exploring them time consuming and complex.

Automating the reuse and the inference of available knowledge is crucial for better decision support, but it remains challenging since machines cannot interpret and understand the meaning of the populated knowledge. Thus, it is important to provide a well-formalized knowledge representing a common understanding of the practices and experiences solving specific domain problems. Knowledge should be represented in a computer interpretable format in order to enable the collaboration of machines with human. Thus, any update within its content or structure is automatically taken into consideration by machines to generate up-to-date decisions. For instance, in healthcare, considerable efforts have been invested in formalizing the medical knowledge and making them available as linked data to be reused by

machines. Bio2RDF database[4] [16] and Linked Life Data[5] are examples of medical knowledge base aggregating and integrating various sources. Nonetheless, the diversity of knowledge representation hinders the smooth integration of existing available knowledge sources to complete each other.

Despite the *integrability*, smart IoT-based systems should be able to solve the domain problems. This requires interacting with domain experts to learn and encode their know-how describing the problems and the solutions. For instance, for the development of knowledge-based clinical decision support system, knowledge should be acquired based on the collaboration with the medical experts to guarantee high quality and provide the right decisions. In this way, it is important to provide a collaborative user-friendly interaction when updating decision rules, since the domain experts are not necessarily familiar with IT. Additionally, providing a flexible and easy to maintain procedural knowledge representation is crucial to facilitate the knowledge update and evolution.

Consequently, the development of smart IoT-based systems identifies a bidirectional knowledge acquisition interaction: (1) from experts to IoT systems where the system is learning from human intelligence to accelerate the decision making; and (2) from IoT systems to experts where hidden knowledge is generated based on analyzing the IoT data. The first interaction relies on acquiring knowledge from experts, while the second interaction requires dealing with big data challenges detailed in the next section.

1.3.2.2 Big Data Challenges

IoT technologies have spawned a growing of the datasets, and have stressed not solely the huge volume of data, but also its diversity and the speed at which it must be managed. Thus, they contribute to creating the big data phenomenon which is defined by the International Data Cooperation (IDC) as "*a new generation of technologies and architectures, designed to economically extract value from very large volumes of a wide variety of data, by enabling high-velocity capture, discovery, and/or analysis*" [17].

However, traditional data-intensive systems and mining algorithms are not able to cope with the scale and highly swift of data, especially unstructured data stemming from multiple wearables and connected things [18]. Taking into consideration the following challenges paves the way for generating ***value*** including new knowledge, insights and recommendations that drive the decision-making process:

- **Data Volume**: According to IDC[6], the volume of digital data is doubling each year, and it is expected to reach 44 zettabytes in 2020. The IoT po-

[4]http://download.bio2rdf.org/release/3/release.html
[5]http://linkedlifedata.com/
[6]http://idcdocserv.com/1678

tentially contributes to increasing the data volume as 32 billion things will be connected by 2020[7]. Collecting and storing this myriad amount of data need to be driven by potential solutions, since data must be available to be exploited by cognitive technologies such as machine learning algorithms in order to create and identify hidden knowledge about the patient, and to visualize complex data for better interpretation. Thus, scalable data storage and analytics should be considered when designing such systems.

- **Data Variety**: Data is structured, semi-structured and unstructured mostly generated by connected devices and mobile systems. As different vendors are investing in IoT market, the data type, units and representation are different with incompatible formats [19]. This heterogeneity leads to the dearth of semantic integration and interoperability which precludes data mining, machine learning algorithms [20], as well as near real-time visualization of the generated data streams. Within these various representations, the meaning of the data is mostly hidden and unspecified. According to IDC [21], in 2013 less than 5% of information was analyzed because very little was known about the data. Failing to take the heterogeneity challenge into account can easily derail the decision making. Thus, data analytic requires content harmonization to provide high quality of data.
- **Data Velocity**: The proliferation of IoT fosters the generation of real-time data that may reach millions of events per second[8]. Thus, timely processing and correlating data stemming from multiple data sources allow portraying more accurate information to provide the right decision at the right time.
- **Data Veracity**: The quality of the data is challenging for accurate analytic, especially in healthcare. For instance, wearable data can be corrupted or imprecise due to failure to wear the sensor [19]. Cleaning the IoT-generated data is required to reject poor quality data and provide better decisions.

1.3.3 System Scalability

The IoT-based system scalability is another important challenge which is adrift from the aforementioned problems. It concerns the ability of the system to support the deployment of new management processes and knowledge/data sources, and to support the storage of the generated big data as well as managing the real-time processing.

In general, the scalability of the systems is constrained with the availability of IT resources including CPU, memory, the storage capacity and the network throughput. In this thesis, we are interested in the scalability from the processes

[7] http://www.emc.com/collateral/analyst-reports/idc-digital-universe-2014.pdf
[8] http://www.gartner.com/newsroom/id/3221818

and the data management perspectives. Details about the mass deployment of IoT devices in the network can be found in [22].

In complex system of systems, large number of management processes should be defined to control the subsystems' evolution. Deploying for each subsystem its own management processes, that allows monitoring the system evolution and taking the decision to adapt its behavior based on context changes, may lead to an overhead and system failure. Moreover, such solution is costly in terms of money since it requires allocating sufficient IT resources for the deployment and the execution of the management processes. Thus, it is important to provide a trade-off between the system scalability and its cost. For instance, sharing and virtualizing the resources based on the cloud computing principles, when deploying management processes, may help reducing the cost.

1.4 Existing Work

Software models and architectures have been introduced to alleviate the system complexity by identifying the main interactions among the system components, while self-managed systems have been enunciated to automate the system management tasks and to minimize the human intervention. Based on the state of the art, we found that the Autonomic Computing initiative [5] has a strong focus on managing complex systems through automating tasks based on the MAPE-K loop pattern (abbreviation of Monitoring, Analysis, Plan, Execution, and Knowledge). Many research activities [23, 24, 25, 26] refer to the autonomic computing for designing adaptive and self-managed systems that automatically adapt their structure and behavior based on the context changes. Adopting the autonomic computing for managing IoT complexity seems promising [27]. Recently, Ben Alaya et al. [22] have proposed the FRAMESELF framework implementing the autonomic computing paradigm for the self-management of M2M systems in the context of smart cities.

Nevertheless, within traditional autonomic and self-adaptive systems, there is insufficient focus on (1) the nature of interactions between the MAPE-K components when multiple loops are managing system of systems, (2) the dynamic coordination among the management processes (Monitoring, Analysis, Plan and Execution) to manage the requirement evolution, and (3) the integration of cognitive capabilities (including the procedural knowledge and human-machine interaction) for generating smart business decision-making [15, 28]. To address this deficiency, we harvested the list of software design patterns, and we surveyed knowledge engineering and management techniques in order to propose *Autonomic Cognitive Design Patterns* that help conceiving *smart IoT-based systems*.

From the platform perspective, to guarantee the horizontal scalability and elas-

ticity, IoT-based systems should leverage big data and cloud computing technologies to store the myriad of data in NoSQL databases and provide flexible data stream processing [29, 30, 31, 32, 33]. However, the heterogeneity of the data remains challenging for data mining and analytics [18]. In this context, we found that ontologies and semantic web are the most used techniques to share a common vocabulary of the domain and enable the interoperability and the integrability among the IoT-based systems [34, 35, 36, 37, 38]. Nonetheless, storing huge amount of data in ontology may lead to inconsistency when maintaining the knowledge structure. Moreover, it may raise scalability challenges due to the frequent access to the ontology when updating its content and reasoning on it [39].

1.5 Thesis Positioning

The main goals of this thesis are (1) enhancing the design and development of complex IoT-based systems through integrating autonomic and cognitive technologies and (2) ensuring the integration of heterogeneous IoT-generated data for better analytic, while guaranteeing the system scalability. To this end, we have classified these challenges into three management levels: *process*, *knowledge* and *data*.

The process level deals with the identification of the management processes' interactions and their coordination to ensure the system functional requirements and manage its context changeability. The knowledge level deals with capturing the experts' knowledge to automate the decision-making, while considering the integration and the reuse of existing external knowledge sources. Finally, the data level deals with huge volume of data, their distribution, velocity and variety to satisfy the system non-functional requirements. With respect to these challenges, we propose an amalgamation of approaches from software engineering and knowledge engineering. Figure 1.2 summarizes the challenges that should be managed as well as the adopted approaches leading to the production of Autonomic and Cognitive IoT-based systems.

For smart IoT manageability satisfaction: We propose a set of *design patterns* that allow the dynamic coordination of the management processes in order to meet the system evolution and solve complex requirements at runtime. The proposed patterns guarantee the *interoperability* through providing a common representation of the data meaning and processes functionalities based on semantic web technologies as well as the *integrability* through connecting the management process and knowledge sources via a mediator. Furthermore, we take into consideration the interaction of IoT systems with human for efficient decision making. Thus, we propose to formalize the procedural knowledge, called also tacit knowledge, based on collaborative technologies that allow acquiring the knowledge from experts for later reuse by the management processes.

For scalability and big data management satisfaction: We propose to integrate the cloud computing to enable the virtualization of resources as well as storing the huge volume of data in cloud clusters. Moreover, we propose to integrate parallel data processing which is suitable to deal with the rapid growth of data.

For generic, evolutive, flexible and extensible system: As technologies continuously evolve, we adopt a model-driven design methodology to have an open, generic and reusable solution that can be instantiated using new technological platforms. Furthermore, we propose to semantically describe the data/knowledge sources and the management processes. Thus, the system is easily extensible at runtime with new sources and processes as long as it understands its meaning and structure.

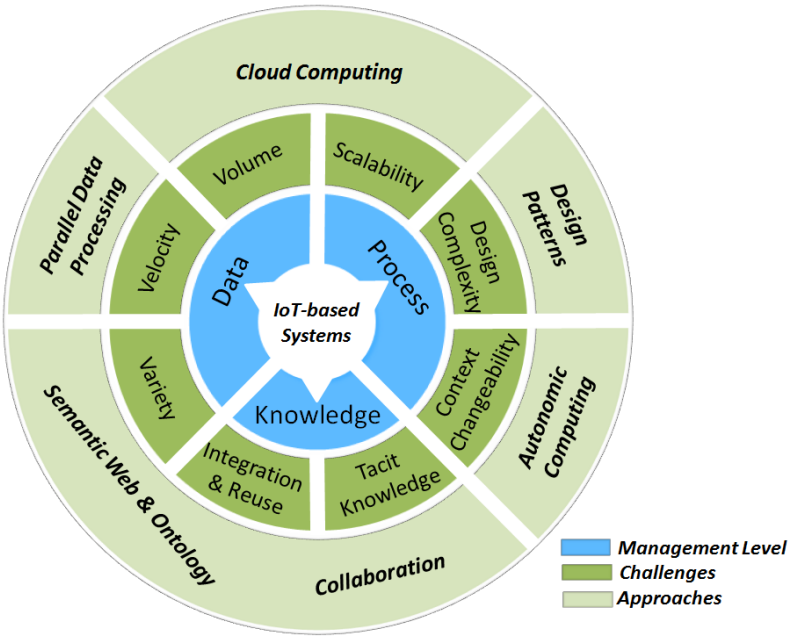

Figure 1.2: Autonomic and cognitive IoT-based systems' challenges and approaches

In the next section, we provide an overview of our contributions and how the existing approaches have been integrated for developing smart IoT-based systems.

1.6 Scientific Contributions

As presented in Figure 1.3, our thesis' contributions fall into three main research directions: *Software Engineering*, *Data & Knowledge Engineering* and *Healthcare* as an applicative domain.

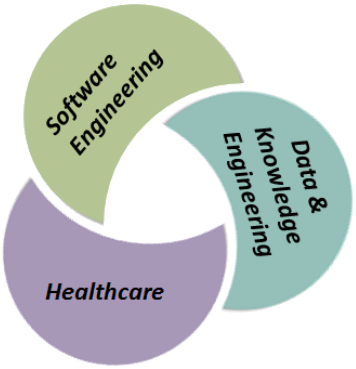

Figure 1.3: Thesis main research areas

Our *first scientific contribution* focuses on the design and coordination of the management processes to meet the IoT-based system requirements evolution. To manage the design complexity, we proposed a *model-driven methodology* that conceptually considers the challenges over the process, knowledge and data management levels. Within this methodology, we defined *a set of design patterns* that propose solutions to: (i) manage the dynamic coordination of the management processes based on the system requirements; (ii) handle the knowledge organization and semantic integration in order to provide smart systems able to think and interact with humans; (iii) and support big data management to provide more accurate insights, while managing system scalability. By following a separation of concerns approach, it is possible to combine the proposed patterns to manage complex system of systems' requirements. Furthermore, to deal with the system' context changeability at runtime, we propose a semantic description of the management processes in order to dynamically discover and activate the appropriate process(es) based on the system evolution. The proposed patterns are domain-independent and formally described with the UML language, which make them reusable and flexible to support domain concepts.

Our *second scientific contribution* proposes a platform specific model hosting the management processes and the generated data. It mainly considers the challenges, raised by the Monitoring and Analysis processes, such as the data heterogeneity and the system scalability as well as the ability to process the data at the right time. To this end, we proposed a *Knowledge as a Service (KaaS)* architecture for enabling the *semantic big data management*. Our KaaS aims at continuously generating new knowledge about the system from the heterogeneous gathered data stored in a cloud environment. It combines both the NIST cloud computing and big data reference architectures, and extends them with a semantic web layer incorporating ontological models describing the system. As an application, we combined and instantiated a set of the proposed design patterns for the development of a *Cog-*

nitive Monitoring System to manage the patient health based on wearables. Thus, we elaborate the *Wearable Healthcare Ontology* for the integration of data stemming from multiple wearable devices; and we deployed the management processes as well as the knowledge/database components on the KaaS. After that, we evaluated the system performance and scalability following different KaaS configuration in order to measure its cost function.

Our *third scientific contribution* enriches our second contribution with new cognitive capabilities in order to allow IoT-based systems thinking and generating decisions concerning the patient treatment. Within this contribution, we mainly focus on the knowledge representation and the plan process for the development of *Prescriptive Cognitive System* able to assist the physicians through automating treatment personalization based on the patient context changes. Medical knowledge is distributed, mostly located in medical experts' minds, which makes sharing and reusing it challenging. To deal with this challenge, we proposed a collaborative methodology for extracting and formalizing the experts' knowledge. The output of this methodology is a fruitful model named *Treatment Plan Ontology (TPO)* which semantically represents the characteristics of the medical interventions. Its flexible structure allows integrating external reliable knowledge sources such as DrugBank. Thus, we proposed and implemented an *ontology-based planning algorithm* that integrates the TPO instances with DrugBank knowledge source to detect drug-drug interactions and provides personalized treatments. Finally, we elaborated two appraisals to evaluate the proposed prescriptive cognitive system from two perspectives:

- *Clinical Evaluation* which refers to the ability of the system to generate the appropriate recommendation. We identified, based on the collaboration of a medical expert from the Hedi Chaker Hospital in Sfax-Tunisia, different real use cases in diabetes type 2 with different complexity levels. Then, we simulated these use cases to compare what our system generates as recommendations to the expert's advice.
- *Performance Evaluation* which refers to the performance of the proposed ontology-based planning algorithm (plan process) in terms of response time and scalability when reasoning. We proposed two approaches to evaluate the planning performance: (1) DrugBank online connection and (2) DrugBank caching system. After selecting the appropriate approach, we evaluated its cost with different cloud configurations to drive the selection of the appropriate one based on the system requirements and associated cost.

1.7 Dissertation Outline

The rest of this dissertation is organized as follows:

Chapter 2. General Background and State of the Art

This chapter provides a detailed understanding of the main concepts which constitute the basis of this thesis' contributions. It is hierarchically structured starting with describing the management processes ending with the most elementary entity which is data. This chapter firstly delineates existing research works dealing with the design of autonomic and self-adaptive systems. Secondly, it details the main approaches used to manage cognitive capabilities including the knowledge and big data layers. Finally, it highlights the cloud computing vision for managing scalability and elasticity. All over these sections, existing works are discussed and some limitations are highlighted.

Chapter 3. Autonomic Cognitive Design Patterns for the Design of Smart IoT-based Systems

This chapter deepens our first contribution dealing with the design of *Autonomic and Cognitive IoT-based systems*. First, it studies existing software design patterns and highlights their reusability to solve problems encountered when designing IoT-based systems. Then, it delineates our proposed model-driven methodology as well as the proposed patterns that enable the dynamic coordination of the management processes, and support cognitive capabilities. The use of each pattern is illustrated with an example from the healthcare domain, more precisely for the patient-centric management.

Chapter 4. A Knowledge as a Service Platform for Heterogeneous Wearable Data Integration

This chapter highlights our second contribution dealing with the semantic integration in the context of big data systems. First, it delineates the use of wearable computing in healthcare. Then, it surveys existing IoT platforms for the management of the generated data, and classifies these works according to three main criteria: semantic management, big data management and cloud management. After that, it provides an overview of the proposed *Knowledge as a Service (KaaS)* platform to manage heterogeneous data based on semantic and cloud-based big data technologies. Through combining a set of the proposed patterns, a *Cognitive Monitoring System* is proposed for managing patient health. Thus, the *Wearable Healthcare Ontology (WH_O)* is elaborated to enable the system understanding the meaning of the received data. The system performance and scalability have been demonstrated in different KaaS configurations.

Chapter 5. A Prescriptive Cognitive System for Patient Treatment Management

This chapter introduces with cognitive capabilities to automate decision making in healthcare. First, it discusses clinical decision support systems and details works supporting the patient treatment adaptation. Then, it provides a *Prescriptive Cognitive System* that enriches the previous system with a formal representation of the procedural medical knowledge describing the medical interventions. This model is named *Treatment Plan Ontology (TPO)* and it is conceived to support the integration of external complementary knowledge sources and acquire decision rules from medical experts. To assist the physicians when taking the decision concerning the patient treatment, an ontology-based planning algorithm has been proposed to reason on TPO and DrugBank in order to generate personalized treatments. The system efficiency is demonstrated through the simulation of real use cases. Likewise, its performance and scalability have been evaluated on the cloud. Finally, recommendations have been proposed to select the appropriate cloud infrastructure based on the system requirements.

Chapter 6. Conclusion & Perspectives

This last chapter summarizes all the work achieved during this thesis, and provides an overview of the perspectives as enhancement of the proposed work.

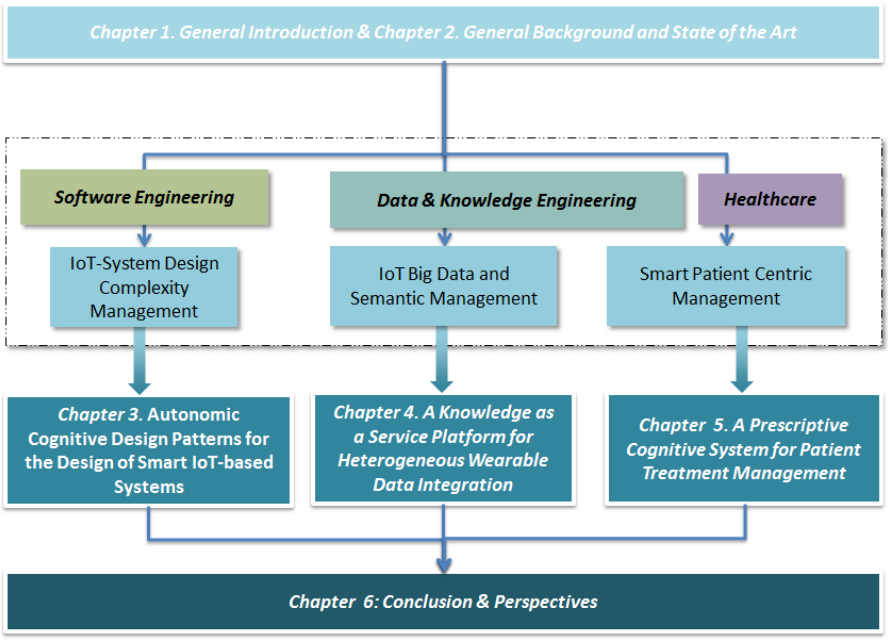

Figure 1.4: Dissertation structure

CHAPTER 2

Background & State of the Art

"If I have seen further, it is by standing on the shoulders of giants."

- Isaac Newton

Contents

2.1 Introduction 15
2.2 Model-driven Methodologies 16
2.3 Autonomic System Design 17
2.3.1 The Autonomic Computing Vision 17
2.3.2 Patterns for the Design of Adaptive and Autonomic Systems 20
2.3.3 Discussion 27
2.4 Cognitive Computing Concepts 28
2.4.1 Knowledge for Cognitive Computing 28
2.4.2 Big Data Management 34
2.4.3 Discussion 37
2.5 Cloud Computing 38
2.5.1 NIST Cloud Computing 38
2.5.2 Knowledge as a Service 39
2.5.3 Discussion 40
2.6 Conclusion 40

2.1 Introduction

With the development of information and communication technologies, systems tend to become more complex to manage. The proliferation of IoT underpins the emergence of heterogeneous System of Systems (SoS) that pinpoint not only the system design complexity, but also managing its evolution.

To tackle the challenges previously mentioned in Chapter 1, several approaches have been proposed in the literature. From the conceptual perspective, model-driven methodologies and design patterns have been introduced in order to mitigate the system complexity and provide flexible and easy to maintain system architecture. From the system management perspective, the autonomic computing has been widely used to reduce human intervention through automating the system management tasks, while the cognitive computing has been introduced for developing smart systems able to solve problems as human do. From the technological perspective, big data and cloud computing have been recently enunciated for managing huge volume of the generated data and the computational load to guarantee system scalability, despite its evolution.

In this chapter, we provide a deep understanding of these approaches and investigate their integration for the development of smart and flexible IoT systems. Thus, we survey existing adaptive and autonomic design patterns for the design of complex systems. After that, we delineate the cognitive computing principals for the development of smart systems from mainly two perspectives: knowledge management and big data management. Third, we provide an insight on the cloud computing and its benefits of IoT systems. Finally, we shed light on new research directions that we followed during the elaboration of this thesis.

2.2 Model-driven Methodologies

Model-driven engineering (MDE) has been introduced to alleviate the system complexity design. Through the abstraction of the physical system, MDE allows the software designer focusing on the relevant details of the system [40]. It emphasizes primarily the use of models that describe complex systems at multiple levels of abstraction and from a variety of perspectives [41].

Software designers should be able to deal with software design problems and provide flexible models with a trade-off between the functional requirements and the non-functional requirements. In this context, design guidelines and best practices (e.g. design patterns) have been proposed to solve design and development problems. Software patterns are particularly relevant to MDE due to the heavy reliance on modeling techniques and principles [42]. Different design patterns including object-oriented design patterns, architectural patterns and workflow patterns [43, 44, 45] have been defined to propose reusable solutions to common design and implementation problems. These patterns identify solutions in different granularity levels describing the interactions among subsystems, activities and objects. For instance, the GoF design patterns [45] have been mainly introduced to solve problems related to object-oriented software design, while the workflow patterns have been proposed to provide solutions when connecting activities to ensure

workflow synchronization [46]. Furthermore, with the advent of new architectural styles such as Service-Oriented Architecture (SOA), Enterprise Architecture (EA) and Cloud Computing (CC), new patterns have been heralded to solve more specific problems [47, 48]. However, these patterns are still informally described, which make reusing and applying them difficult.

Basically, model-driven engineering including design patterns are relevant for the design of complex IoT-based systems [49]. Recently, a new project named Papyrus for IoT[1] proposed a model-driven development environment for managing heterogeneous applications, provide real-time models and design methods describing interconnected devices and monitoring critical system. Nonetheless, due to the dynamic evolution of IoT systems and the heterogeneity of the generated data, which is the fuel of business insights, providing flexible IoT systems is challenging. Consequently, proposing models that deal with the IoT challenges from the data, processes and infrastructure perspectives increases the productivity and facilitate the system maintenance and extensibility. Moreover, designing IoT-based systems should not be limited to the identification of the components interactions, but also integrate self-management mechanisms that facilitated the system management with minimal human interventions. For instance, systems implementing real-world applications continuously evolve and churn out large amount of heterogeneous data which make their management tricky.

In the next section, we provide details about a well-known paradigm that has been used to enable the self-management capabilities, called autonomic computing.

2.3 Autonomic System Design

In this section, we first provide an overview of the autonomic computing; then, we survey existing autonomic and self-adaptive design patterns for the design of self-managed systems.

2.3.1 The Autonomic Computing Vision

Originally inspired from the human autonomic nervous system and its ability to react to changes, the autonomic computing [5] has been enunciated to overcome the management complexity and to reduce its cost. Despite overwhelming the system administrators with maintenance tasks and managing distributed nodes, the autonomic computing defines self-management mechanisms that aim at automating these tasks based on the system context changes, with minimum human interventions.

[1] https://www.eclipse.org/community/eclipse_newsletter/2016/april/article3.php

The development of autonomic systems for IT organizations follows an evolutionary approach that identifies five maturity levels starting from the basic level ending with the most high level which is the autonomic level [50, 26]. More the management functions are automated, more the system maturity level is increased.

- *Basic Level*: It is the starting level where the system management is manually done. Human skills are required for monitoring, analyzing, taking decisions and executing them.
- *Managed Level*: At this level, new technologies are introduced to automatically collect data about the system and its environment. Human skills are required to analyze the collected data and take the decision for the system management.
- Predictive Level: It is an evolution of the managed level where the system is able to analyze the collected data, identify the symptoms and predict future performance. At this level, the human efforts are reduced, but their intervention is still required to approve (or not) the recommendation and to initiate the corrective actions.
- *Adaptive Level*: At this level, the closed loop is automatically executed. The system is able to automatically monitor, analyze, decide and execute the actions based on the provided knowledge. The IT-management is driven by business policies such as Service Level Agreements (SLA), response time, etc.
- *Autonomic Level*: It is the most mature level. The business policies and objectives govern the IT infrastructure operations. At this level, the system is able to understand the business metrics, optimize e-business performance and quickly deploy newly optimized e-business solutions. Human concentrates on the business level and interacts with the autonomic system to monitor the business processes.

An autonomic system corresponds to a managed element which is monitored and controlled by autonomic manager(s) through touchpoints: sensors and effectors (see Figure 2.1). An autonomic manager implements the MAPE-K pattern that includes a Knowledge block and the Monitor-Analyze-Plan-Execute functional composition. The interactions among these components have the overarching goal to enable the self-managing behavior.

The *Monitoring* collects data about the managed element(s) and its external environment through sensors, while the *Analysis* identifies a request change and sends it to the *Planning* component, if an abnormal situation has been detected. If it is the case, the Planning reasons on the knowledge to select the appropriate plan, and sends it to the *Execution* component. This later performs the selected plan through

effectors. All these autonomic components operate on the *Knowledge*, which constitutes the foremost element that facilitates automating the system management tasks.

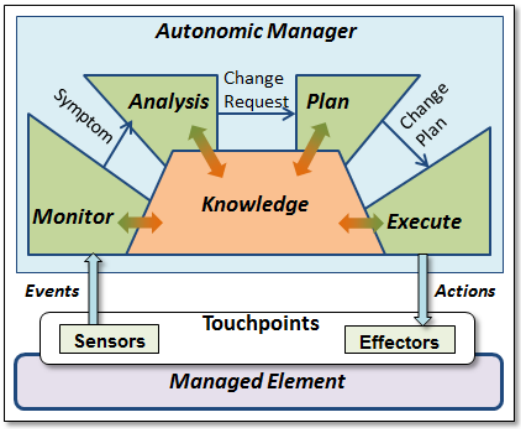

Figure 2.1: The autonomic computing reference architecture

Many research works have adopted the autonomic computing paradigm for the development of self-managed systems [24, 51, 52]. Managing autonomic systems implementing real-world applications with a single MAPE loop is challenging [28], since unforeseen requirements may appear at runtime [15]. Thus, conceiving multiple control loops that coordinate their management functions and delegate tasks to each other to solve more complex situations is required.

However, the lack of design guidelines describing the autonomic components coordination yields to a high design and implementation cost since conventional autonomic systems are considered as ad hoc solutions – designed and implemented from scratch [53, 54]. Additionally, new data capture technologies such as wearables and connected things pave the way for the development of advanced systems with business benefits, but create new challenges such as heterogeneity and scalability. Hence, the emergence of new business requirements may lead to re-engineering the system to integrate new management processes to attain self-management properties. Indeed, design models that describe the structure and the behavior of the management processes remain unclear to the autonomic system designers who should conceive flexible systems that effectively operate to guarantee functional and non-functional requirements.

In the next section, we provide a thorough survey of the existing research activities that proposed design patterns for building autonomic systems and modeling the interactions among the management functions.

2.3.2 Patterns for the Design of Adaptive and Autonomic Systems

Autonomic and self-adaptive research community has spent little efforts on the concrete implementation, the interaction and the coordination of autonomic components [15, 53]. We surveyed the literature covering research works proposing design patterns and models for building autonomic systems. Table 2.1 detailed the harvestable works from three perspectives: the management processes interactions, the knowledge and data management, and the principals used for the software design.

Dazzi et al. [53] proposed a behavioral pattern based on the GoF strategy pattern for the design of autonomic stream-classification-systems. This pattern aims at enabling the dynamic reconfiguration of the classifier behavior according to the input stream. However, the proposed pattern is specific to the stream classification and does not consider the heterogeneity of the input stream for the reconfiguration (such as format, meaning, etc.). Furthermore, it does not support the design of complex system, where different autonomic components should be coordinated. Solomon et al. [55] proposed an autonomic architecture based on real-time patterns for automating the system IT management. The main focus of this work is dealing with problems from the communication perspective such as synchronous or asynchronous messages exchange, and concurrency management. Based on a separation of concerns approach, the authors identified the main autonomic system components and their basic functionalities that can be deployed in distributed nodes to lighten the computational workload. These components are orchestrated based on the coordinator component. Nevertheless, the coordination is limited to the components belonging to one loop managing one system.

Based on the generalization of several existing GoF design patterns [45], Ramirez and Cheng [54] proposed twelve adaptation-oriented design patterns that aim at separating the development of the functional logic from the adaptive logic. The proposed patterns are formalized using UML class and sequence diagrams to describe the objects within the monitoring, the decision-making, and the reconfiguration processes (adding, removing, and reconfiguring a server at runtime). The authors extended the template description with behavior and constraints-based fields to enables developers to understand the consequences and trade-offs. In the same context, Mannava and Ramesh [56] proposed a pattern, which is the combination of existing design patterns (see Table 2.1), for designing dynamically reconfigurable systems. They are mainly interested in providing a solution that blocks all transactions and communications among the components until the system is successfully configured. Thus, they avoid the system deadlock state when inserting, removing or modifying of the component parameter at runtime. Frey et al. [57] proposed a set of architectural design patterns for integrating autonomic systems. The inte-

gration may occur “internally" when integrating MAPE loops within one system or “externally" when integrating MAPE loops managing independent systems to form a coherent system. The authors focused on solving the possible goals conflicts that may appear within the autonomic integration and presented seven patterns illustrated through use cases of smart homes connected to a micro smart grid.

However, none of the aforementioned works delineate the interaction and coordination of multiple autonomic loops to manage complex systems. In this context, Al-Shishtawy et al. [58] proposed four patterns for the coordination of multiple autonomic managers for the network management:

- the *stigmergy* pattern represents indirect interaction where the autonomic managers make changes on the managed resources, and these changes are sensed by other autonomic managers that will do more actions. Within the stigmergy, undesired behaviour may occur at runtime if many autonomic managers are involved.
- the *hierarchical management* pattern adopts a reflexive approach in which the autonomic managers (children) monitors and manages the systems. In turns these managers are controlled and orchestrated by another autonomic manager (parent). This pattern may cause scalability problems in complex systems, when the parent should manage a big number of autonomic managers.
- the *direct interaction* pattern relies on binding to the appropriate managers.
- the *sharing of the managed elements* pattern refers to sharing information about their states (knowledge) and synchronizing their actions.

However, these patterns are abstract since the interactions among the MAPE-K functions are not clearly presented. Recently, Weyns et al. [28] presented a seminal work identifying the interactions of MAPE loops through considering both the inter/intra-interactions. The authors proposed five decentralized design patterns for self-adaptive systems. The first two patterns are fully distributed while the rest of the patterns are based on hierarchical approach that reflects a separation of concerns among the control loops. Choosing the right pattern or combination of patterns depends on the system requirements and its complexity.

- Within the *coordinated control* pattern, autonomic components belonging to the same autonomic manager can interact with external autonomic components having the same type and belonging to another autonomic manager. In other words, this pattern describes the M-M, A-A, P-P and E-E interactions.
- The *information sharing* pattern illustrates the M-M interaction, while the other components operate independently. From the scalability perspective,

this pattern provides more scalable solution than the first one, since it limits the interactions to M-M.

- The *Master/Slave* pattern considers the (M, E) as slaves specific to the managed element, while the (A-P) as master shared among the slaves. For large-scale systems, this pattern may cause problems related to the master overhead, if many Monitoring components send data to the Analysis component. Moreover, the master remains the failure point.
- The *regional planning* pattern proposes for each region one Plan component which is shared among the (M, A, E) of each sub-system belonging to this region. This Plan may interact with another Plan belonging to another region.
- Finally, the *hierarchical control* pattern which is similar to the hierarchical management pattern presented by Al-Shishtawy et al. [58].

Both Al-Shishtawy et al. [58] and Weyns et al. [28] proposed patterns for the coordination of autonomic managers. However, they did not consider the communication and data integration challenges as well as the system requirements evolution at run-time when new requirements appear (e.g. the deployment of new autonomic managers that should interact with existing ones). Frey et al. [57] proposed a set of architectural design patterns for integrating autonomic systems. The integration may occur “internally" when integrating MAPE loops within one system or “externally" when integrating MAPE loops managing independent systems to form a coherent system. The authors focused on solving the possible goals conflicts that may appear within the autonomic integration and presented seven patterns illustrated through use cases of smart homes connected to a micro smart grid. Recently, Abuseta and Swesi [59] proposed a metamodel describing the MAPE-K loop interactions for designing self-adaptive systems. The proposed pattern covers the M-M, M-A, A-A, A-P, P-P, P-E and E-E interactions based on the observer pattern. They used the class diagram to model the structural patterns as well as the sequence diagram to model the behavioral patterns. The authors instantiated the proposed patterns to manage the QoS (the scalability, the response time and throughput) of a Virtual Learning Environment via enabling the load balancing adaptation.

To ensure the coordination of the management processes at runtime, Oliveira et al. [60] provided an autonomic coordination based on the knowledge component and exchanging events among the monitoring and execution components. They identified two types of knowledge: the private knowledge which is shared with the components of the same loop, and the public knowledge which is shared among the existing MAPE loops. To synchronize the execution of the communicating control loops, the authors proposed to share and exchange a token among all the loops. Thus, each loop needs to request for the token, and when this latter is released the first loop requesting the token starts its execution, and so on.

	Management Processes		**Knowledge & Data Management**	**Software Design Principals**			
Related Work	*Basic Interaction/ Coordination*	*Self-Provisioning*		*Separation of Concerns*	*System Requirements*	*Reused Patterns*	*Patterns Modeling*
Dazzi et al. [53]	-	-	-	-	stream-classifier	Strategy Pattern	UML Class Diagram
Solomon et al. [55]	Coordination of autonomic blocks belonging to a single loop deployed in a distributed way	-	-	Yes: Data Acquisition, Filtering, Decision Making, Coordinator	Reusability, Reliability, Data integrity, Concurrency	Filter Chain Pattern; Real-time pattern; Guarded Call (syn); Message queue (asyn); Mutual exclusion semaphore	UML Component Diagram
Ramirez et Chen [54]	Not detailed	-	-	Separate the adaptive from the functional logic	IT management Response time	GoF Pattern	UML Class & sequence diagrams

	Management Processes		Knowledge & Data Management	Software Design Principals			
Related Work	*Basic Interaction/ Coordination*	*Self-Provisioning*		*Separation of Concerns*	*System Requirements*	*Reused Patterns*	*Patterns Modeling*
Mannava and Ramesh [56]	Basic Interaction	-	-	-	Web service composition	Worker Object Aspect Oriented, Lookup, Row Data Gateway, Observer, Strategy, Case Reasoning	UML Class diagram
Weyns et al. [28]	5 patterns for the basic Interaction of autonomic managers	-	-	Yes	Scalability, Confidentiality	-	Abstract representation
Abuseta and Swesi [59]	Basic interaction through the Observer	-	-	Separation of adaptation logic from functional logic	QoS Management: Scalability, Response time, Throughput	Observer Pattern	UML Class & Sequence Diagrams

	Management Processes		**Knowledge & Data Management**	**Software Design Principals**			
Related Work	*Basic Interaction/ Coordination*	*Self-Provisioning*		*Separation of Concerns*	*System Requirements*	*Reused Patterns*	*Patterns Modeling*
Oliveira et al. [60]	Interaction & coordination based on event communication	-	Public knowledge for synchronization & Private knowledge	Separate platform from Infrastructure	Scalability, Stability, Concurrency, Confidentiality	-	Abstract representation
Frey et al. [57]	Internal & external interactions	-	-	Yes	Scalability, safety, robustness, evolvability, conceivability	-	Abstract representation
Vidal et al. [49]	MAPE basic Interactions (Event-based)	-	Data heterogeneity is partially considered through characterizing the sensor measurements	Yes: Each MAPE function is responsible of atomic function	Real-time processing, loosely coupled	publish-subscribe	UML Class Diagram

	Management Processes		Knowledge & Data Management	Software Design Principals			
Related Work	*Basic Interaction/ Coordination*	*Self-Provisioning*		*Separation of Concerns*	*System Requirements*	*Reused Patterns*	*Patterns Modeling*
Al-Shishtawy et al. [58]	4 abstract patterns for the basic Interaction of autonomic managers	-	-	Yes: Functional decomposition	-	-	Abstract representation

Table 2.1: A survey of works dealing with design patterns for autonomic and self-adaptive systems

The solution proposed in [60] is very efficient to answer confidential and concurrency issues and enable the system stability. It doesn't impact the system scalability but highly impact on the processing time. The authors applied the proposed solution to the cloud computing in order to coordinate the adaptation of both the PaaS and IaaS according to the workload, the VM charge fees and the physical machine utilization. However, locking the execution of the loop excludes the parallel management and increases the waiting time of the adaptation, which is not adapted for real time system management.

Based on this survey, we noted that none of these patterns integrate the IoT systems, except the work of Vidal et al. [49] which presents a model-driven methodology, named MindCPS, for the design and development of autonomic Cyber Physics System (CPS). The methodology implements a metamodel that drives the system design through defining the interaction of the autonomic components and the main functionalities. The proposed model describes the sensors measurements and protocols, identifies simple and complex filters to process the received data and detects symptoms to enable the adaptation. A set of model-to-code transformation, which automatically generates the JAVA code implementing the autonomic control loop as well as EPL queries for the real-time management of the events and SQL queries for the management of the non-real time data, are identified. However, the proposed model did not take into consideration the data heterogeneity and its volume, as well as the scalability of such complex system.

2.3.3 Discussion

Despite the diversity of the proposed research activities, there are open challenges [15, 61] that should be considered when designing autonomic systems such as:

- the coordination of decentralized MAPE functions (management processes),
- the distribution and the heterogeneity of the generated data,
- the interaction with the knowledge which remains the cloudy component in autonomic computing,
- and the self-provisioning of the management process in order to manage at runtime the system context evolution which is unpredictable at design time.

As advocated in Table 2.1, almost works detail the MAPE interactions at design time except the work of Solomon et al. [55] which coordinates the autonomic components belonging to the same control loop, and the work of Oliveira et al. [60] which coordinates different autonomic managers based on shared public knowledge. Moreover, none of the presented works provides a solution that enables the dynamic discovery of the management processes to meet the system requirements and context evolution at runtime. With the proliferation of IoT, most of the data are

unstructured generated by sensors with different representations. Thus, new patterns that extend the basic management processes to support processing and storing big data are required.

Conventional autonomic and adaptive patterns do not portray the knowledge structure and its interaction with the MAPE functions. Indeed, autonomic computing technologies do not rely on instructive and procedural information [4] which are the basis for smart decision-making. In the basic autonomic computing model, knowledge is considered as "*almost any sort of structured data or information that could be used to carry out processes, especially processes that can be automated*"[2]. For the development of smart autonomic IoT system, knowledge should not be overlooked. It is not limited to describing tasks for automating system management, but it should include cognitive capabilities that allow IoT systems learning and generating business decisions to interact with human.

In the next section, we provide an insight on the cognitive computing concepts that can serve for the smart management of autonomic IoT systems.

2.4 Cognitive Computing Concepts

Wang [4] defined the Cognitive Computing as "*an emerging paradigm of intelligent computing methodologies and systems that implements computational intelligence by autonomous inferences and perceptions mimicking the mechanisms of the brain*". This paradigm is the amalgamation of concepts from artificial intelligence, natural language processing, ontologies, and big data analytics [62] for the development of smart systems. In this section, we are interested in the knowledge representation based on ontologies as well as big data technologies for the design of cognitive systems.

2.4.1 Knowledge for Cognitive Computing

Cognitive computing is not limited to machine learning; it includes also the knowledge modeling and ontologies construction[3]. But first, it is important to distinguish knowledge from data and information, especially with the emergence of the IoT and mobile systems that instantly generate huge volume of raw data. The DIKW pyramid has been introduced to hierarchically organize data, information, knowledge and wisdom [63]. As presented in Figure 2.2, data is at the bottom. It represents primary data gathered from sources, such as IoT, unprocessed and unorganized facts. Information, next pyramid layer, is data processed, organized and associated to the context. Knowledge represents the practical use of the processed

[2]http://www.ibm.com/developerworks/autonomic/library/ac-edge6/

[3]http://www.dataversity.net/cognitive-computing-semantic-technology-worlds-connect/

information, or put into action. And on the top level of the pyramid, wisdom is the application of knowledge resulting in the ability to add value and provide insights for decision-making.

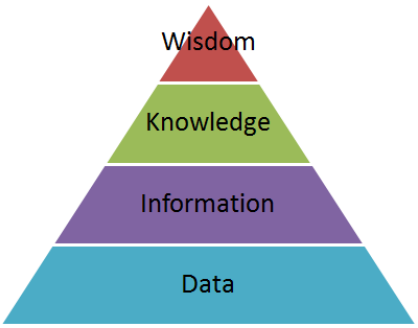

Figure 2.2: DIKW pyramid

Davenport and Prusak agreed with this distinction and defined the knowledge as "*a fluid mix of framed experience, values, contextual information, and expert insight that provides a framework for evaluating and incorporating new experiences and information*" [64]. Many classifications of the knowledge type have been proposed:

From the *cognitive psychology* area, John Anderson, a psychology professor at Carnegie-Mellon University, introduced two classifications for the knowledge type: *declarative* and *procedural* [65]. *Declarative* knowledge consists in describing the know-that including facts and information about things, while the *procedural* knowledge focuses on delineating the know-how including skills and steps of a task or procedure.

From the *knowledge management* research area, knowledge can be *explicit* or *tacit* [63]. *Explicit* knowledge can be easily accessed, transferred and shared, since it is published in multimedia content taking the form of textual, image or video representation. Contrarily to the explicit knowledge, the *tacit* knowledge, which refers to the know-how, is critical to understand and reuse since it is located in the human heads and embedded in human behavior. While explicit knowledge can be straightforwardly integrated into knowledge management software systems, tacit knowledge must first be converted to explicit knowledge.

Woods and Cortada [66] summarize these definitions as follows: the declarative is viewed as explicit and ties to "describing", while the procedural is viewed as tacit and ties to "doing".

2.4.1.1 Knowledge Management

Practically speaking, the value of the knowledge leans on the way it is managed and explored. In this context, a set of knowledge management activities (KM) have been proposed to represent, capture, understand, share and reuse the knowledge through which the decision is made. Figure 2.3 portrays the main KM activities that foster the knowledge extraction and application based on IT technologies:

- *Knowledge Acquisition*: It represents the most important process where the needed knowledge to perform a task is extracted. Traditionally, it was viewed as the process of extracting the knowledge from experts and transferring the knowledge into knowledge based system.
- *Knowledge Storage*: It refers to the process of storing the knowledge in repositories for present and future use [67]. Recent IT technologies promote the storage to guarantee the accessibility to the knowledge.
- *Knowledge Sharing*: Once the knowledge is stored, it should be shared to enable the collaboration that increases the productivity. Despite its importance in KM, many cultural and social factors may hinder knowledge exchange and reuse.
- *Knowledge Reuse*: It encompasses three roles [68]: knowledge producer who generates the knowledge, knowledge intermediary who indexes and categorizes the knowledge and knowledge consumer who retrieves and uses the knowledge to create value.

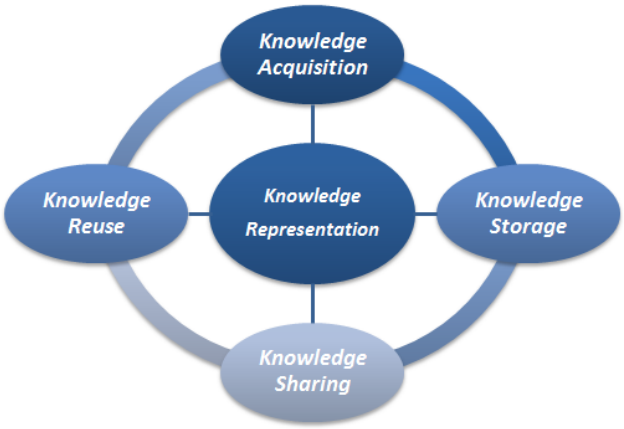

Figure 2.3: Main activities of knowledge management

To enable the system learning and thinking using inference and reasoning techniques, knowledge should be well-formalized and represented based on a common understanding in order to allow the system interpreting the meaning behind

it. James Kobielu, the IBM Big Data evangelist, said: "*For cognitive computing to achieve its promise we need a thick metadata layer that incorporate semantic tagging formats*"[4]. Thus, it is important to provide semantic support for the development of smart systems, because data is nothing without meaning.

2.4.1.2 Ontologies for Knowledge Representation

In Artificial Intelligence, Knowledge Representation (KR) is the fundamental process that aims at transforming explicit knowledge into computer-interpretable format in order to automate the reasoning process. Among the first KR languages that have been proposed, we cite the Frame [69] and the Semantic Networks [70]. However, these languages lack formal (logic-based) semantics [71]. To overcome this deficiency, the Description Logic (DL) was introduced. In this context, Sowa [72] defines the knowledge representation as "*the application of logic and ontology to the task of constructing computable models for some domain*". Based on first order logic, DL guarantees a formal expressiveness of the knowledge representation [73] and can be used to describe domain knowledge through using ontology.

Ontology is the key for offering smart support for KM by highlighting the meaning of the silos of information and representing the knowledge in a computer interpretable format able to be understood, despite its heterogeneity [74]. Originally, ontology is defined as "*a branch of metaphysics relating to the nature and relations of being*" [5] . It tends to describe what exists. In computer science, Gruber defines the ontology as "*an explicit specification of a conceptualization*" [75]. It describes the semantics of the data based on a common shared understanding of the domain through classes, relations and axioms that add meaning to information, and provides techniques that automatically seek for the required information [76, 77].

For the development of smart IoT systems, ontologies can be used from two perspectives:

- From one hand, they can be used to formalize the declarative knowledge and provide a common understanding of the system context. For instance, M2M and IoT-based systems leverage this technique to describe the sensors, their observations and their properties. We cite the Semantic Sensor Network Ontology (SSN) [78] proposed by the W3C as the most popular ontology in this domain. Moreover, they have been considered as a main solution for integrating heterogeneous systems and data sources [79, 80].

- From the other hand, ontologies can be used to describe the problems and actions for the decision making -the procedural knowledge. Thus, IoT systems may reason and aggregate this knowledge to provide decisions based

[4]http://www.dataversity.net/cognitive-computing-semantic-technology-worlds-connect/
[5]Merriam-Webster: http://www.merriam-webster.com/dictionary/ontology

on the observations they capture. For example, in healthcare, ontologies have been used to describe clinical guidelines in order to develop knowledge-based clinical decision support systems.

Regardless of the purpose of use, ontology representation requires following a methodology for building its schema and capturing the real-world. In the next section, we delineate the main methodologies for formalizing the knowledge based on ontologies.

2.4.1.3 Methodologies for Building Ontologies

Many methodologies have been proposed to build ontologies [81, 82]. METHONTOLOGY is the most known methodology developed in the Artificial Intelligence Lab from the Technical University of Madrid (UPM). Its skeleton was developed through taking the IEEE 1074-1995 standard as a starting point [83]. It covers the ontology development process, the ontology life cycle and the techniques used to achieve each activity [84] from the specification to the maintenance of the implemented ontology. On-To-Knowledge [85] is another methodology that introduces the balance between human problem solving and the automated IT solutions, since knowledge management is not governed only by IT. It starts with a feasibility study where problems and actors are identified. Then, the ontology requirements are specified, and a semi-formal ontology description of the target application is provided. Based on the first model, a refinement process is initiated to produce an application-oriented ontology. After that, the provided ontology is evaluated by checking the requirements. Finally, the ontology is applied in the application environment and maintained.

METHONTOLOGY and On-To-Knowledge methodologies do not consider collaborative construction of ontologies [86] which is a complicated task and mandatory to formalize the experts' tacit knowledge (procedural knowledge) and converge to a unified ontology. In this context, Kotis and Vouros [87] proposed a Human-Centered Ontology Engineering Methodology (HCOME). Unlike the existing methodologies, the HCOME contributes to involving the knowledge worker in the ontology development. HCOME starts by identifying the requirements and the aim of the ontology by discussing with collaborators. Then, each worker will focus on developing his/her ontology in his/her personal space by importing ontologies, consulting top ontologies and discussing with experts. At this stage, each worker manages his/her own ontology versions. These ontologies are shared with other workers or collaborators for review and evaluation. Thus, this methodology requires a deep knowledge in ontology formalization and modeling, however, domain experts generally are not necessary familiar with ontology development. In the same context, NeOn methodology [88] identifies nine scenarios for collaboratively building ontologies and ontology networks, reusing and re-engineering

knowledge resources (ontological and non-ontological). However, NeOn does not define a workflow for the ontology development based on human involvement for tacit knowledge conceptualization.

Formalizing the tacit knowledge into an ontological model relies on the experts' collaboration as the key success. Despite the advantages of IT to create a sharable and virtual environment that overcomes space and time constraints, the face-to-face communication remains more effective and exhibits higher performance results [89]. Thus, considering social barriers and personality traits in the methodology processes is required, which is not well-addressed in the previous works, in order to foster knowledge sharing, collaborative ontology construction and acquisition. Indeed, studies in psychology have shown that knowledge sharing behavior among individuals is influenced by personality traits [90].

2.4.1.4 Semantic Web for Cognitive Computing

Once the ontology structure is defined, it should be implemented in order to be shared and reused during the annotation process. Semantic web is well-known for these purposes. It has been introduced by Tim Berners-Lee in 1998 as "*an extension of the current web, in which information is given well-defined meaning, better enabling computers and people to work in cooperation*" [91]. Based on standardized languages such as OWL, RDF and RDFS recommended by the World Wide Web Consortium (W3C), semantic web data can describe the knowledge content underlying both web pages and multimedia content like images and videos [92].

Many semantic web technologies have been proposed and evaluated to store large scale annotation following the RDF format [93]. Other semantic web technologies have been developed to enable the collaboration among human and computers such as the semantic wikis. Semantic wikis [94] represent an evolution of basic wikis, which are the most popular web-based solution that enables human-human collaboration and exchanging the knowledge through acquiring new contents from several users in the web [95]. Through coupling ontologies with wiki-based platforms, semantic wikis unleash the power of both technologies to smartly manage the knowledge and to guarantee accessing to data from anywhere at any time. Many semantic wiki engines [96] have been proposed as open source or commercial platforms with different characteristics and purposes. They have been classified into two categories [97]: wiki for semantics that supports collaborative ontology development such as Semantic MediaWiki [98] and semantics for wiki that focuses on adding meta-data to wiki pages such as SweetWiki [99].

2.4.2 Big Data Management

Big data is attracting more and more interests from the research and industries through its ability to provide new discoveries and insights. There is no a complete agreement on big data definition, some definitions associate it to the data distribution and scalability management, while other to the massively parallel data processing. The National Institute of Standards and Technology (NIST) defined the big data as follows [100]: "*Big Data consists of extensive datasets–primarily in the characteristics of volume, variety, velocity, and/or variability–that require a scalable architecture for efficient storage, manipulation, and analysis*". From this definition, managing big data refers to deal with two-pronged: data storage, and data analytic and visualization.

2.4.2.1 Big Data Storage

Nowadays, the digital universe accounts more than 90% of unstructured data with different format such as text files and multimedia contents [17]. Surely, IoT systems point out this phenomenon through generating dynamic and heterogeneous data. According to the EMC[6], the percentage of data generated by mobile "things" will grow to 27% in 2020. Due to the limited processing speed and the significant storage expansion cost [101], conventional relation databases, which are originally conceived to manage structured data, are inadequate to store IoT-generated data. For instance, fed by sensors and connected things, real-time IoT-applications should support fast read and write operations within large data sets in order to provide timely information.

Bearing that in mind, the NoSQL databases have been developed to deal with such limitations. The main purpose of NoSQL is to offer flexible management of distributed non-relational data models, guarantee the horizontal scale over servers nodes and provide high availability of the data [102, 103]. Four NoSQL data storage approaches have been identified [104]: key-value database, document database, column-family database and the graph database. For instance, MongoDB, which is a document database, has been used to store IoT-generated data [101, 105]. Coupling NoSQL technologies with the cloud computing [106] is a suitable solution for the management of IoT data in order to guarantee the scalability of the system.

Steadily storing IoT-generated data without analyzing them is a waste of resources. Big data is not limited to storing large amount of data, but it includes another important phase which is data analytics.

[6] http://www.emc.com/leadership/digital-universe/2014iview/internet-of-things.htm

2.4.2.2 Big Data Analytics & Visualization

More type and larger-scale the data are, the higher accuracy the analytics results will have [107]. However, the diversity of the data format, its huge volume and its quality represent barrieres for better analytics. Traditional analytics tools, including data mining, machine learning and statistical tools, are not able to cope with both the scale and the high interference of IoT-data [18]. In this context, big data analytics bring advanced and rigorous techniques that deal with the processing and the analysis of distributed, heterogeneous and huge amounts of data [108, 20, 109]. Thus, new patterns can be identified to retrieve hidden information and provide straight business insights for smarter and faster decision-making. Given this background, new data analytic platforms are required to parallelize the processing of the data being generated.

Recently, research activities in healthcare have been invested to leverage the potential benefits of big data. DiabeticLink[7] is an example of big data platform that has been developed by the University of Arizona and National Taiwan University for patient empowerment and personalization [110]. It allows exchanging disease information and experience through aggregating multiple sources of data such as forums, drug side effects, and electronic health records. The genomic research area has also benefit from big data [111] to discover novel genes and help personalizing the treatment. We cite for instance the collaboration that has been made between Cloudera and the Institute for Genomics and Multiscale Biology to provide a big data platform that assists researchers predicting and understanding the treatment of disease[8].

Technically speaking, many platforms and libraries have been developed to deal with the big data analytics. Apache Hadoop[9] is the first popular open source platform for batch processing based on the MapReduce framework operating on Hadoop Distributed File System (HDFS) or on other external databases. It supports the parallel processing of large-scale data stored in distributed server nodes. Hadoop has been widely used in industry [112], for instance Yahoo uses Hadoop running over 42,000 severs at four data centers for searching and spam filtering, while Facebook announced that their Hadoop cluster is able to process 100 petabyte (PB) data.

Presently, many plugins have been developed over Hadoop to extend its capabilities such as Pig Latin developed by Yahoo Research that offers a high level language for expressing data analysis programs for encoding data analysis tasks in MapReduce programs; and Hive initially introduced by Facebook for analyzing its

[7] http://www.diabeticlink.org/

[8] http://www.zdnet.com/article/cloudera-and-mount-sinai-the-structure-of-a-big-data-revolution/

[9] http://hadoop.apache.org/

generated data and for generating reports [113]. Other Apache projects have been developed for data collection and ingestion such as Sqoop[10] that allows importing relational databases such as MySQL into the HDFS; Flume[11] that allows aggregating and moving large amounts of log data; and Chukwa[12] that allows collecting large scale distributed data and rapidly processing it. Other projects have been introduced to manage the performance of the big data platforms and coordinate the parallel processing such as Oozie[13] and Zookeeper[14]. Recently, Apache Spark[15] has been developed for the large-scale data processing based on the MapReduce framework. It is considered as 100 times faster than the Hadoop MapReduce in memory.

To enable more complex analytic such as clustering and predictive analytic, new platforms offering parallel data mining and machine learning algorithms operating on large-scale data are required. In this context, Mahout[16], ML-Hadoop[17] and Spark MLlib[18] libraries have been developed to offer scalable machine learning. Freshly, Apache Flink[19], which is an open source platform for distributed stream and batch data processing, has introduced FlinkML[20] library implementing supervised machine learning and recommendation algorithms.

The strengths of big data resides also on its ability to provide real-time big data analytics which is defined as "*the ability to make better decisions and take meaningful actions at the right time*" [114]. To this end, distributed publish/subscribe platforms have been developed to support the integration of other big data stream processing platforms. For instance, LinkedIn has developed Apache Kafka[21] for messaging and has integrated it with Apache Samza[22] for distributed stream processing. Apache Storm[23] is another interesting framework for stream processing. It is a distributed fault-tolerant real-time computation system. Apache Storm is scalable and allows processing until one million messages per second per node. Also, Apache Spark introduces the spark streaming library which has been used by Netflix[24] associated with Kafka to consume events from Netflix devices, while Apache

[10] http://sqoop.apache.org/
[11] http://flume.apache.org/
[12] https://chukwa.apache.org/docs/r0.3.0/index.html
[13] https://oozie.apache.org/
[14] http://zookeeper.apache.org/
[15] https://spark.apache.org/
[16] https://mahout.apache.org/
[17] https://code.google.com/p/ml-hadoop/
[18] http://spark.apache.org/docs/latest/mllib-guide.html
[19] http://flink.apache.org/
[20] https://ci.apache.org/projects/flink/flink-docs-master/apis/batch/libs/ml/index.html
[21] http://kafka.apache.org/
[22] http://samza.apache.org/
[23] http://storm.apache.org/
[24] https://spark-summit.org/2015/events/spark-and-spark-streaming-at-netflix/

Flink introduces Flink DataStream API[25] which has been used by Bouygues Telecom[26] to provide real-time diagnostics and alarming based on users upstream.

Despite big data analytics, data visualization plays an important role to interact with experts and accelerate the decision-making through mapping the analytic results into interactive multimedia contents (images, videos, etc.). Some visualization libraries have been developed such as D3.js, Polymaps, NodeXL[27], etc. For instance, to visualize twitter conversations, the Moebio Labs[28] developed the Newk[29] application.

2.4.3 Discussion

Cognitive computing feeds on big data and leverages the knowledge modeling to provide smart systems. The main purpose of big data analytic (batch and stream processing) is to retrieve more accurate results and help scientist and domain experts to discover and extract new information not known in advance. Despite this wide range of big data platforms and libraries, the lack of data semantic description impedes data analytic. The concern is not related to having more data, but is more related to the ability of the system to understand the meaning of the data in order to be correctly processed and aggregated with existing structured/unstructured databases. An IoT big data system should be able to curate the data and understand its meaning in order to be automatically processed and to get better decisions. These aspects should be considered since the design of the system in order to provide flexibility and extensibility when adding new data sources that enrich the analytics.

Semantic web contributes to the development of cognitive computing through giving meaning to the data emanating from heterogeneous and distributed data sources, and annotating the extracted information for further reuse. Moreover, data integration is crucial, especially in IoT systems where different vendors are manufacturing devices implementing different syntax and formats. Cleaning and filtering the received data from IoT systems remain a mandatory task to provide a high quality of data. Likewise, it is important to guarantee the availability of data and knowledge for real-time analytics and reasoning.

[25] https://ci.apache.org/projects/flink/flink-docs-master/apis/streaming/index.html
[26] http://data-artisans.com/flink-at-bouygues-html/
[27] http://www.nodexlgraphgallery.org/Pages/Default.aspx
[28] http://moebio.com/
[29] http://moebio.com/newk/twitter/

2.5 Cloud Computing

Cloud computing is a new paradigm, well known for its ability to deliver highly scalable distributed computing platforms in which computational resources are offered as a service [115]. It provides advanced mechanisms for data storage, computation, and dynamic resource allocation according to real-time computation needs. The cloud computing is characterized by the elasticity concept which refers to the ability to add or remove resources in order to manage the system workload much more closely [116].

2.5.1 NIST Cloud Computing

The NIST[30] defines the cloud computing as "*a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [117]. The NIST has defined a generic cloud computing reference architecture that regroups various cloud services into three service models: infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) [118]. The IaaS layer contains the physical resources, such as servers, processors, and networks, on which the platforms will be deployed. The PaaS layer offers platforms for data storage, programming languages, and web application servers. Finally, the SaaS layer handles software applications that directly access infrastructure resources or refer to the PaaS layer for their computing platform and data access.

Recent works [119, 120] focus on integrating of the cloud computing and IoT systems in order to leverage storing the data in cloud [106], processing the data [33] and efficiently sharing resources based on virtualization to manage the system cost [121]. This combination opens the door for developing scalable system ensuring the near real-time data processing in order to generate new knowledge about the system at the right time. Making this knowledge reusable, accessible and available is crucial for the development of smart IoT systems.

Many research activities took advantage of the cloud and its ability to offer everything as a service to propose new cloud layers such as Thing as a Service [122], Big Data as a Service [123], Sensor Data as a service [124] and Machine Learning as a Service [125], etc. In the next section, we detail works dealing with Knowledge as a Service.

[30] NIST: National Institute of Standards and Technology

2.5.2 Knowledge as a Service

Several works adopted the KaaS in different domains to facilitate sharing and accessing knowledge from different sources. Knowledge as a Service was defined by Xu et al. [126] as "*the process by which a knowledge service provider answers queries presented by knowledge consumers through a knowledge server*". Knowledge is typically extracted from large volume of data coming from heterogeneous data owners according to knowledge models and is then delivered as a cloud computing service. Based on these knowledge models, the knowledge server is able to deliver the right answer to the right consumer at the right time [127]. Figure 2.4 represents an overview of the KaaS paradigm.

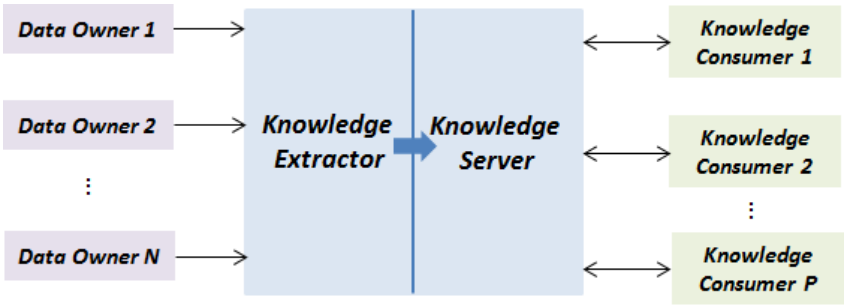

Figure 2.4: The KaaS paradigm

Qirui [128] brought new thinking to agricultural information-system development by using the KaaS approach. In this approach, KaaS provides services that offer recommendations about planting on the farm according to user specifications and environmental factors. The knowledge representation in this KaaS is based on ontologies, while the data are stored in a relational database (MySQL). Kannimuthu et al. [129] applied KaaS in the e-commerce domain, where they focused mainly on how to extract knowledge from data based on data mining techniques to attract the user to other products of the same enterprise. Ultimately, this leads to financial benefit for the enterprise. In their approach, data are formatted and stored in an XML database. Another interesting approach was proposed by Lino et al. [130], who used KaaS to facilitate emergency response in natural disasters like tsunamis and earthquakes using interactive digital TV. In this work, knowledge is shared by means of ontological descriptions. Moreover, the authors focused on implementing a planning algorithm for emergency response in the KaaS layer to support evacuation of unsafe areas. However, the system architecture seems to be restricted to a specific client/server architecture as opposed to an accepted cloud computing architecture.

2.5.3 Discussion

The KaaS approach seems to be a very interesting concept that can serve the IoT-based systems, since these systems churn out mountain of data that should be automatically processed to generate new knowledge (business benefits). The generated knowledge should be accessible and available for better decision making. From the literature, we noted that Qirui [128], Kannimuthu et al. [129], and Lino et al. [130] all proposed a KaaS architecture; however, they did not follow a well-accepted cloud computing reference model such as those proposed by NIST, CISCO, or IBM. Moreover, none of the aforementioned works did consider managing big data to generate new knowledge from distributed heterogeneous data sources, especially in case of IoT-based systems which are data-intensives, complex and distributed systems. This huge volume of data should be managed with appropriate methods and techniques in order to generate more accurate and precise analytics.

2.6 Conclusion

In this chapter, we provided an overview of the main research areas to which this thesis belongs. We delineated also existing research works dealing with the challenges presented in chapter 1 and we identified the techniques and approaches they are using. Moreover, we highlighted the use of existing approaches for the design and development of smart IoT systems and pointed out how their contributions are synergistic. To summary, based on studying existing works, we notice that there is:

- Lack of guidelines and patterns that drive the design of smart IoT systems while taking into consideration the dynamic evolution of the system context, big data and scalability challenges.
- Lack of knowledge and big data management to support the smart management of IoT systems through enabling cognitive capabilities.

Consequently, the aim of this thesis is proposing solutions that foster the development of ***Autonomic and Cognitive IoT systems***. The next chapter will detail our first contribution dealing with managing the design complexity of such systems.

CHAPTER 3

Autonomic Cognitive Design Patterns for the Design of Smart IoT-based Systems

"Simple things should be simple and complex things should be possible."

- Alan Kay

Contents

3.1 Introduction 42

3.2 IoT Healthcare System for Patient Treatment Management . . . 42

3.2.1 Autonomic Computing for Patient Treatment Management . 43

3.2.2 Maturity Levels for Autonomic and Cognitive IoT-based Systems 44

3.3 Design Patterns Identification 46

3.3.1 Management Processes Level 46

3.3.2 Data Management Level 47

3.3.3 Infrastructure Management Level 50

3.4 A Model-driven Methodology for the Design of Autonomic and Cognitive IoT-based Systems 50

3.4.1 An Overview of the Proposed Methodology 50

3.4.2 Management Processes Coordination Patterns 52

3.4.3 Semantic Knowledge Mediator Pattern 71

3.4.4 Big Data & Scalability Management Patterns 73

3.5 Conclusion 79

3.1 Introduction

IoT systems implementing real-world applications continuously evolve. To deal with this dynamic evolution, management processes are required in order to automatically manage the system requirements, while reducing the human efforts and the associated cost. An IoT-based system represents an example of complex system of systems where the coordination of the management processes is needed to guarantee their smooth functioning. However, IoT-based systems are inherently distributed, heterogeneous and complex to design and manage.

Consequently, in this chapter, we identify the problems that may occur when designing smart IoT-based systems, we study in the literature existing reusable software patterns and we highlight their limitations. After that, we propose a model-driven methodology to assist the architect designing and developing smart IoT-based systems. Based on the autonomic and cognitive computing approaches, we define within this methodology a set of autonomic cognitive design patterns to iteratively drive the architect when identifying the interaction among the management processes. The proposed patterns integrate existing design patterns, and extend them with new capabilities to support (1) the dynamic coordination of the management processes and (2) big data management, while guaranteeing a flexible and extensible architecture. We refer to the healthcare as an applicative domain of IoT-based systems. Thus, we illustrate the efficiency of the proposed patterns when managing the patient treatment. But first, we delineate the utility of autonomic computing in healthcare and its ability to automate the patient treatment management based on medical sensors and wearable devices in order to accelerate the decision-making and avoid health complications.

3.2 IoT Healthcare System for Patient Treatment Management

The human is a complex system of systems composed of interconnected and dependent organs and cells which are coordinated to work together in order to regulate the body's internal mechanisms. Thus, the dysfunction of one sub-system caused by a chronic disease and/or the application of a medical intervention to manage a problem may affect other sub-systems. For example, the management of a patient with type 2 diabetes should consider the management of the heart failure risk factor [131], since some diabetes drugs may be linked to heart failure[1]. Consequently, it is important to coordinate the management of different sub-systems and provide a preventive approach when adjusting the patient treatments.

[1] https://www.nlm.nih.gov/medlineplus/news/fullstory_158144.html

3.2.1 Autonomic Computing for Patient Treatment Management

We propose to adopt the autonomic computing paradigm (1) to reduce the complexity of managing the treatment of patients with chronic diseases, and (2) to expedite the decision-making based on IoT devices. The ultimate goal is to enable the dynamic prediction and/or detection of the patient health deterioration as well as proposing personalized treatment plan, while taking into consideration the patient health evolution. Figure 3.1 portrays the management processes that can be automated for the patient treatment management based on the autonomic computing. For clarity reasons, we present only one loop managing a chronic disease. These processes, forming the MAPE loop, operate on the *Knowledge* which is the foremost element that enables the coordination of the MAPE processes for the smart management of the patient treatment.

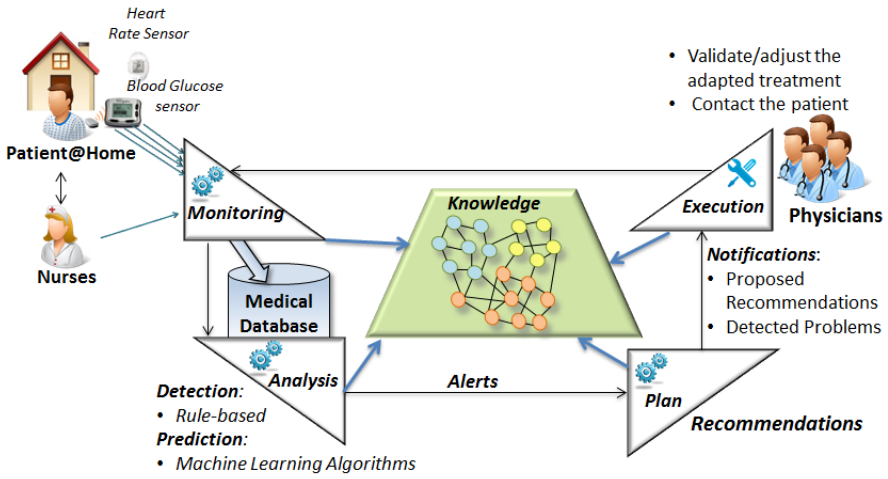

Figure 3.1: Autonomic computing for managing patient treatment

- The *Monitoring* process collects the patient data coming from medical sensors and wearable devices, and/or measured by health professionals.
- The *Analysis* process analyzes the received data to identify patient health deterioration. This later may implement detection rules, or predictive algorithms such as machine learning. According to the severity of the detected or predicted anomaly, an alert will be sent to the Plan process.
- The *Plan* reasons on the knowledge to automatically search the appropriate treatments based on the patient medical conditions, the interventions contraindications, the medical interactions and the drug side effects. Finally, the personalized recommendations are sent to the physicians for validation.

- When managing the patient treatment, the *Execution* process requires the physician intervention to approve the execution of the plan.

When managing patient treatment, complex situations such as comorbidity may occur. Comorbidity refers to the simultaneous or sequential occurrence of two disorders or illnesses in the same person. It also implies interactions between the illnesses that affect the course and prognosis of both [132]. In such cases, multiple management processes should be deployed and coordinated in order to manage the presented diseases and avoid possible complications derived from the diseases' interactions.

3.2.2 Maturity Levels for Autonomic and Cognitive IoT-based Systems

Automating the combination of the management processes depends on the patient context and the system requirements, while their interactions may reflect different maturity levels of the system. We identified four maturity levels, as portrayed in Figure 3.2, which combine the management processes of the autonomic computing with cognitive capabilities to represent the timeline evolution of the autonomic and cognitive IoT-based systems development. We specialized these maturity levels for the smart management of the patient treatment, based on IoT systems. The proposed maturity levels are the following:

- The *Cognitive Monitoring Management* level is the basic level that allows IoT system interacting with human through collecting and visualizing the observations. At this level, only the monitoring process is automated and it implements cognitive capabilities that allow perceiving the received data streams. This level is adequate for near real-time visualization of the system context evolution. For example, in healthcare, it is important to continuously monitor the glucose level of prediabetes or elderly people that have the risk to get diabetes, and automatically detect possible degradation.
- The *Predictive Cognitive Management* level is an evolution of the *Cognitive Monitoring Management* level. At this level, the system goes further the visualization and detection to apply intelligent mechanisms such as machine learning and data mining algorithms that allow the system learning and predicting other related anomalies initially imperceptible from existing observable parameters. For example, if the patient has confirmed diabetes and is following specific treatments to manage her glucose level, it is important to predict the hypertension as it is a risk factor of diabetes, especially if the patient is not equipped with a blood pressure sensor.

- At the *Prescriptive Cognitive Management* level, the system is able to provide business decisions to the expert based on the system context and through reasoning on the procedural knowledge, which includes the business rules populated by the business experts. As decisions are business related, the system sends the recommendations to the appropriate practitioner. For example, if the system detects that a patient with diabetes is getting worse while following a specific treatment; it will adapt the treatment through generating another personalized treatment and send it to the right physician.
- The most mature level is the *Autonomic Cognitive Management* level. In system of systems, the context is dynamically changing, especially if the subsystems are interconnected and dependent. Thus, the system should be able to dynamically discover management processes based on the sub-systems' context evolution in order to provide a proactive management. For example, we consider a patient is managed through (M_d, A_d, P_d, E_d) deployed for diabetes management. Because of aging, the patient may develop hypertension, thus, it is important that the system should be able to automatically search and activate the appropriate management processes managing the hypertension disease, besides diabetes, while interacting with experts who validate their activation as well as their recommendations for automatic execution.

Maturity Level	*Cognitive Monitoring Management*	*Predictive Cognitive Management*	*Prescriptive Cognitive Management*	*Autonomic Cognitive Management*
Capabilities	** Monitor the business context, ** Detect degradations, ** Visualization	** Predict the context evolution based on Machine Learning Algorithm	** Provide personalized business recommendations, ** Reasoning and Thinking	** Self-provisioning of the autonomic processes to manage the context evolution, ** Reasoning and Thinking
Automated Processes	** Monitoring	** Monitoring, Analysis	** Monitoring, Analysis, Plan	** Monitoring, Analysis, Plan, Execution ** Dynamic discovery of Processes
Human Role	** Analysis and, ** Take decisions	** Validation of the Analysis, ** Take decisions	** Populate Procedural Knowledge ** Recommendation Approval/Modification	** Populate Business Rules ** Recommendation Approval/Modification

Figure 3.2: Autonomic cognitive IoT-based system maturity levels

Reaching these maturity levels requires managing many challenges starting from the dynamic coordination of the management processes ending with the man-

agement of the generated data (volume, velocity and variety). Consequently, in the next section, we identify the main design and development problems that we have faced when building smart IoT systems, identify useful reusable patterns in software engineering, and highlight their consequences.

3.3 Design Patterns Identification

From the survey that we have conducted in chapter 2, section 2.3.2, we found that the patterns proposed by Weyns et al. [28] are the most relevant for the design of autonomic systems. However, these patterns remain abstract and do not detail the following points:

- the knowledge component and its interaction with the management processes which explicitly increases the complexity of the patterns descriptions.
- the coordination of the management processes as well as the dynamic discovery of these processes to automatically meet the context evolution.
- And finally, how to deal with challenges related to the integration of IoT like big data management, and how to perceive the received data for the development of cognitive IoT-based systems.

Consequently, we identified and classified the encountered problems into three categories related to (1) the *management processes*; (2) the *data management*; and (3) the *infrastructure management*. We gathered from the literature useful patterns that may cover the identified design problems. Table 3.1 summarizes our findings and elucidates the list of patterns that can be reused.

3.3.1 Management Processes Level

Generally, the coordination of the management processes remains an open issue in autonomic cognitive management. We found that in Artificial Intelligence domain, the blackboard pattern [133] has been widely used for the dynamic control and coordination of the knowledge sources (KS) based on a control component (C). The blackboard pattern provides effective solution to the design and implementation of complex systems where heterogeneous modules have to be dynamically combined to solve a problem [133]. The control component supervises the shared blackboard among the KS. If the blackboard is modified, the control component activates the appropriate KS. It seems that this pattern is useful to model the management processes coordination, since we consider them as knowledge sources that generate new knowledge concerning the managed element.

Moreover, in complex systems, multiple management processes should be deployed to manage several sub-systems at the same time. The distribution of the

management processes allows gaining performance management, but their interactions become more difficult. Within a distributed system, management processes may not have enough information concerning other separate processes such as their definition and endpoint in order to interact with. In this context, we found that the mediator pattern [45] provides a loosely coupled solution where the distributed management processes may communicate. Nevertheless, the heterogeneity of the processes description is challenging to ensure the interoperability. Thus, we propose to extend the mediator with a common semantic description of the processes to overcome such challenge.

3.3.2 Data Management Level

The IoT market witnesses an important increase. Thus, managing the system scalability is crucial to provide timely information, particularly when consuming the data generated by such devices. The publish/subscribe pattern [134] offers a scalable management of the monitoring process as well as an easy extensibility of the system to support new devices. However, the publish/subscribe pattern presents inflexible semantic coupling. Indeed, in this pattern, the semantic and data structure are well defined. Thus, if a new device is deployed to send different data structure (e.g. different data ordering and data separator), the consumer fails processing the data. With this in mind, we propose to provide a flexible solution by enriching the publish/subscribe pattern with a semantic description of the data structure and ordering. Thus, adding new devices will not affect the data consumption process, since the consumer knows all information related to the received data: the meaning and the structure. Consequently, we propose to use ontology as a semantic description method to deal with the heterogeneity from two perspectives: first to describe the management processes, their functions and endpoints; second to describe the meaning of the managed data and its structure to provide more accurate analysis.

Another important aspect that characterizes the IoT-based system is the near-real time data processing to provide timely information. With the variety of the received data, it is important to filter and curate the data in order to easily integrate and aggregate it, and provide near real-time visualization. Coupling the interceptor pattern with the chain of responsibility pattern [45] provides a solution for near-real time data curation based on filters implemented in the subscriber. Thus, the development of filters based on the semantic description seems promising.

Problem Description	*Pattern Name / Approach*	*Consequences*
Management Processes		
Managing complex autonomic systems requires coordinating various management processes in order to deal with the evolution of the context changes.	Blackboard Pattern	** Dynamic control/ changeability ** Reusability ** Robustness
Management processes are distributed and have little or no knowledge of the definitions of other separate processes that may be reused to deal with context evolution.	Mediator Pattern	** Loosely coupled communication ** Maintainability
Management processes are heterogeneous, which make reusing them difficult and requires programming efforts for the integration and their composition.	Semantic description of the management processes (Ontologies)	** Interoperability ** Dynamic discovery of the management processes ** Maintainability
The number of IoT devices keeps growing rapidly, which may cause scalability problems when consuming the data.	Publish/subscribe Pattern	** Scalability ** Extensibility
Data Management		
Based on connected things, the monitoring process generates huge volume of data stored in distributed clusters that should be processed. Data velocity: data stream processing	Master/Slave Pattern	** Parallel data Processing ** Performance ** Response time ** Fault-Tolerant

IoT devices generate heterogeneous stream data with different representations, which make its processing at real-time is a tricky task, especially that each vendor uses its own format. Sharing a common description and harmonizing the data to get accurate analysis are required.	Semantic description of data structure and meaning (Ontologies) Interceptor & Chain of Responsibility Patterns	** Sharing common sense ** Data integration ** Data Filtering & Curation
Existing IoT-based systems may store the generated data in distributed external databases. Each database has its own structure and its own API to extract data, which may increase the maintainability cost.	Adaptor Pattern	** Reusability
Both the Monitoring and Analysis Processes need to frequently access to the knowledge to extract the semantic and understand the meaning of the received data. In complex systems, where multiple management processes are deployed, this causes performance problems.	Cache Pattern	** Performance ** Scalability
In dynamic systems, the knowledge is dynamically updated, which requires notifying the caches with the changes in order to allow the management processes processing the updated information.	Observers Pattern	** One-to-many dependency management
Infrastructure Management		
Limitation of the resources to deploy the management processes and the knowledge component.	Single Root I/O Virtualization Pattern	** Elasticity
In complex systems, deploying multiple management processes increases the cost. Sharing these processes promotes cost management.	Multi-tenant Pattern	** Sharing the cost among consumers.

Table 3.1: Inventory of reusable patterns

The semantic description including ontologies plays an important role in providing cognitive systems able to perceive and understand the data. However, in large-scale systems, the frequent access to this layer may cause bottleneck and performance limitation, especially in distributed systems. In this sense, we found that the cache pattern[2] is useful. Thus, management processes that frequently require understanding the data meaning should have their own cache. Nonetheless, the update of the cache is challenging, especially in highly dynamic systems where new devices are added and the context is changing. At this point, we found that the observer pattern [45] can be reused to automatically update the distributed caches at the right time, thus, provide accurate autonomic management.

3.3.3 Infrastructure Management Level

Deploying a high number of management processes and knowledge components to manage complex systems is constrained by the availability of IT resources. Increasing the IT resources for the system function leads to an increase of the cost management. Cloud computing comes with solutions to deal with such problems. Virtualizing the resources will gain the system elasticity and the ability to automatically allocate the resources based on the process workload. Moreover, conceiving a multi-tenant architecture, where the management processes and the knowledge components share the IT resources with respect to the confidentiality, fosters sharing the cost among the consumers, thus, reducing the cost management.

Based on this study, we propose an amalgamation of the surveyed patterns and herald new patterns for the design of autonomic and cognitive IoT-based systems. These patterns will be enunciated within a model-driven methodology that elucidates the different phases for the design of smart IoT-based systems.

3.4 A Model-driven Methodology for the Design of Autonomic and Cognitive IoT-based Systems

We present, in this section, an overview of the proposed methodology as well as the proposed patterns. We illustrated the use of these patterns when managing the patient treatment.

3.4.1 An Overview of the Proposed Methodology

We propose a collaborative model-driven methodology that combines software modeling and knowledge engineering principals to facilitate the design and de-

[2] https://msdn.microsoft.com/fr-fr/library/dn589799.aspx

velopment of autonomic and cognitive IoT-based systems. Figure 3.3 depicts an overview of the proposed methodology. Two main phases are identified: (1) *Requirements Identification* and (2) *Requirements Formalization*.

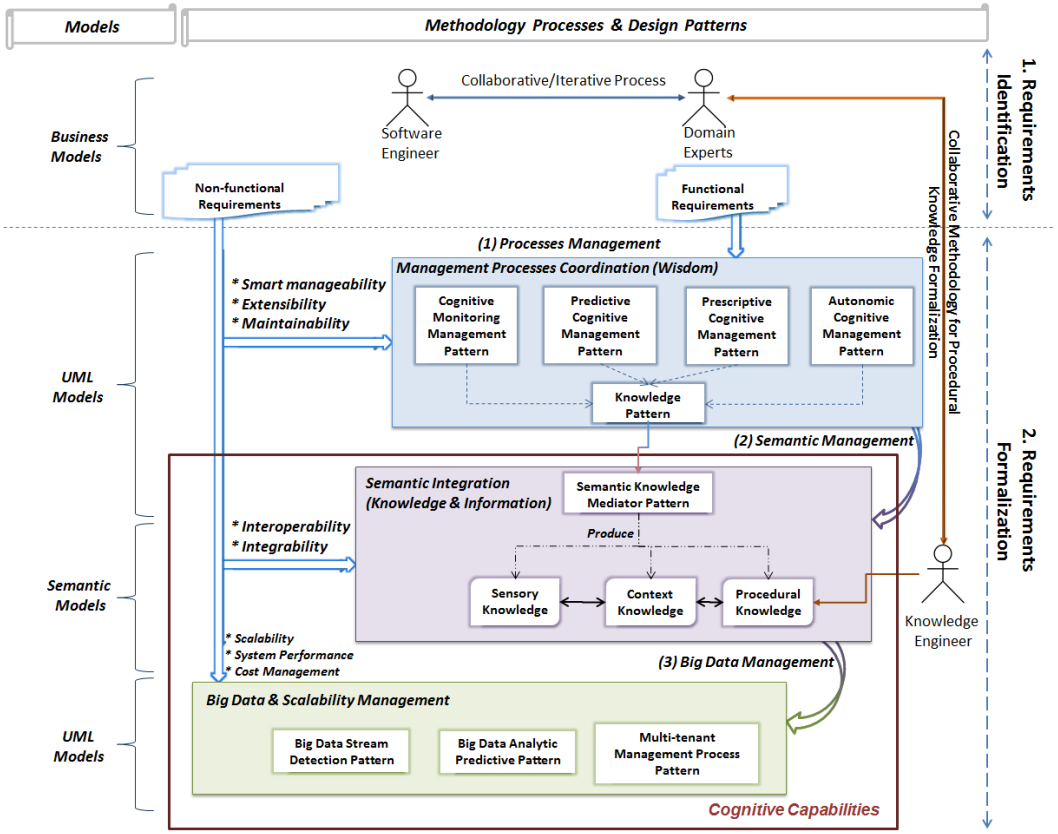

Figure 3.3: A model-driven methodology for the design of autonomic and cognitive IoT-based systems

Requirements identification is based on discussions with the domain experts in order to retrieve the system functions and identify the non-functional requirements. It is an iterative process, where the functional requirements are incrementally refined and represented using business models describing the behaviour of the system without specifying any implementation details.

The next phase, which is the *Requirement Formalization*, focuses on formalizing and structuring the identified requirements into concrete models describing the system processes' interactions. The aim of our methodology is providing smart IoT-based systems, thus, we propose to map the system functions into management processes which can be the monitoring, analysis, plan and execution process; or their combination. For the development of smart IoT-based systems, we denote the existence of a new actor, in the second phase, who is the the knowledge engineer.

This later collaborates with the domain experts in order to formalize the domain knowledge (business problem-solving knowledge) in order to acquire the system the ability to generate business decisions. Within the *Requirement Formalization* phase, we introduce three sub-levels in order to incrementally deal with challenges related to the design of smart IoT-based systems such as the coordination of management processes, the semantic integration and the big data management:

- The first level is the "*Management Processes' Coordination*". Within this level, we identified five patterns that consider the smart manageability, extensibility and maintainability of IoT-based systems. These patterns delineate how the management processes should be coordinated and interact to meet the system's functional requirements based on the knowledge pattern. The proposed patterns identified also the system maturity level and have been conceived to be combined to solve complex requirements.
- Once the management processes are identified and modeled, the next level is the "*Semantic Integration*" where mainly the information about the system and its environment as well as the procedural knowledge (know-how) for decision-making are formalized in order to be automatically reused by the management processes. Within this level, we identified the "*Semantic Knowledge Mediator*" pattern to guarantee the interoperability and the integration for the smooth functioning of the system. The application of this pattern leads to the production of three types of knowledge: the sensory, the context and the procedural knowledge. It is worth mentioning that the knowledge engineer intervenes at this level and collaborates with the domain experts when formalizing their tacit knowledge (know-how).
- Finally, the last level is the "*Big Data & Scalability Management*" where we defined three patterns that deal with the big data challenges (volume, variety, velocity), scalability, system performance as well as the system cost management.

In the following, we deepen the proposed patterns, which have been generalized for the design of autonomic and cognitive IoT-based systems; and we illustrate their use through examples from the healthcare domain.

3.4.2 Management Processes Coordination Patterns

We proceed with a separation of concerns approach where the functional requirements are decomposed into atomic requirements ascribed to the appropriate atomic management process (business logic) that should be coordinated to manage complex requirements. To facilitate the system design, we propose four patterns that

model the management processes coordination, and a knowledge pattern that identifies the main knowledge types that will interact with the management processes. These patterns refer to the defined maturity levels (section 3.2.2), and they can be combined to manage more complex situations and manage the context evolution based on the system context and requirements.

To this end, we referred to the blackboard pattern as the basis for the coordination. To clearly illustrate the use of these patterns, we abstract away both the knowledge and control components in the examples, while we delineate them in the pattern model.

3.4.2.1 Knowledge Pattern

How the knowledge component is organized for the smart management of IoT-based systems?

Context. In general, an autonomic system that manages the business level and provides decisions concerning the managed element requires the declarative knowledge -detailed information about the system and its environment (know-that), and the procedural knowledge -deep business knowledge (know-how) to take decisions.

Problem. Many works dealing with the autonomic computing centralize the knowledge component and provide a common repository describing the knowledge. However, there is a dearth in defining how the management processes interact with the knowledge. In general, all the management processes are connected to a single repository which may cause a bottleneck and scalability problems. Moreover, mixing everything together causes problems when maintaining the knowledge.

Solution. To deal with these challenges, we propose to organize the knowledge into sub-components that will be reused later to identify the interactions with the management processes. This decomposition deals with separation of concerns within the knowledge, and has been inspired from the human multi-store memory model that includes the sensory memory, the short-memory and the long-term memory [135]. More precisely, we referred to the Tulving classification [136] which identifies within the long-term memory three types of memories *episodic memory*, *semantic memory* and *procedural memory*. The *episodic* contains experiences and events in the context in which they occurred, while the *semantic* is more structured and includes facts, meanings and general knowledge about the world by referring to concepts shared with others independent from the personal context. Both *semantic* and *episodic* memories constitute the declarative memory [137]. Meanwhile, the procedural memory includes expertise, skills and procedures "Know-how".

Given this background, we propose three types of knowledge which are the followings:

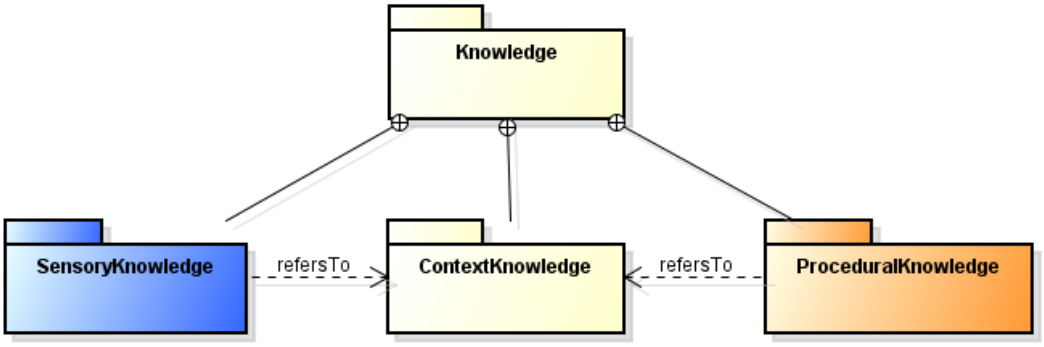

Figure 3.4: Knowledge pattern

- **Sensory Knowledge**: [Sensory Memory + Semantic Memory]. It describes the monitored data and the main characteristics of the sensors used to capture the data (such as the name of the parameter, the units, its endpoint, etc.). It is worth mentioning that IoT generated data is not stored in the Sensory Knowledge.
- **Context Knowledge**: [Episodic Memory + Semantic Memory]. It describes the managed element experiences including the conditions, the context evolution and events.
- **Procedural Knowledge**: [Procedural Memory + Semantic Memory]. It describes the know-how that includes the skills and solutions that can be performed to solve problems. It is acquired from the domain experts based on their practices or learned from past experiences.

As advocated, these knowledge components are interconnected through concepts and facts represented through a common semantic vocabulary describing the system context.

Example. Consider an IoT-based system integrating autonomic computing to adapt the patient treatment (i.e. automatically monitoring the patient health indicators based on sensors, detecting the patient health deterioration and recommending the appropriate decisions to the physicians). To ensure such functionalities, the system should be able to understand the meaning of the received data and explore the patient medical conditions and history as well as the disease management strategies to provide the right decisions to physicians. Figure 3.5 instantiates the proposed knowledge pattern for the patient treatment management based on IoT devices.

Consequence: The decomposition of the knowledge structure fosters the collaboration when acquiring its content and facilitates the maintenance process of each sub-component. Moreover, by distributing these components, the system offers better scalability. However, the distribution may raise problems when main-

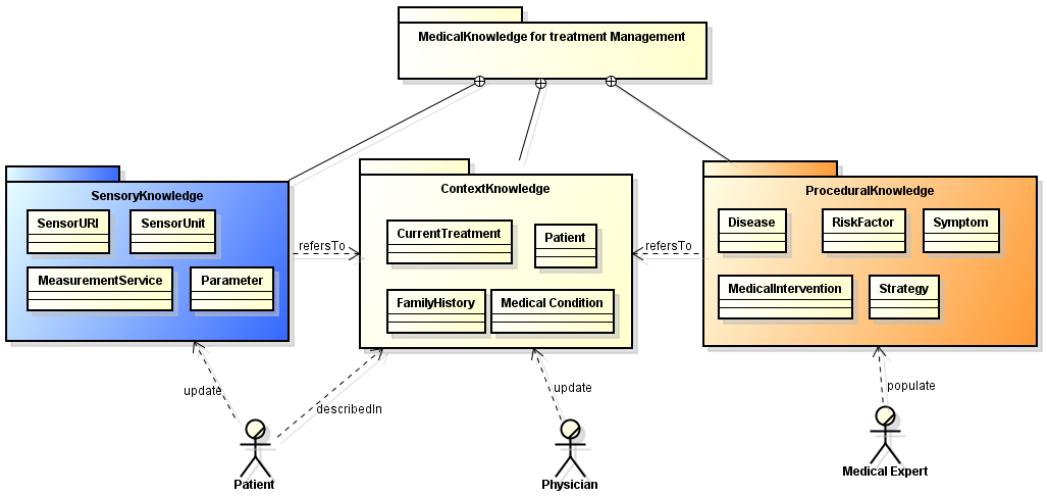

Figure 3.5: Instantiation of the knowledge pattern in healthcare

taining the classes that interconnect these knowledge sources. As a solution to this problem, we propose to provide a generic schema for each knowledge sub-component, then, it will be specialized to the domain. Thus, the integration of these knowledge sources is based on the generic concepts.

3.4.2.2 Cognitive Monitoring Management Pattern

How the monitoring process can perceive the data and interact with experts for IoT-based system management?

Context. The integration of IoT devices promotes the development of context-aware applications. Through collecting real-world data, the experts/users may get more information about the system evolution. A smart IoT-based system should be aware of the used devices and understand the meaning of the received data to detect context changes, and offers visualization services.

Problem. Conventional monitoring systems are developed for specific devices generating data with the same unit, using the same syntax and representation. If new IoT devices are integrated, new applications need to be developed to explore the acquired data which is costly. Moreover, there is a lack of bi-directional interaction between IoT systems and human, since the primary objective was getting the captured data.

Solution. We propose to apply the Blackboard pattern in order to enable the dynamic interaction of IoT systems with the experts/users. Figure 3.6 represents a conceptual model designing the interaction between IoT systems and human. We identified two types of interactions: the *IoT-Human* interaction to visualize the data and receive notifications in case of context changes, and the *Human-IoT* interaction

to manage the system through modifying its context and configuration.

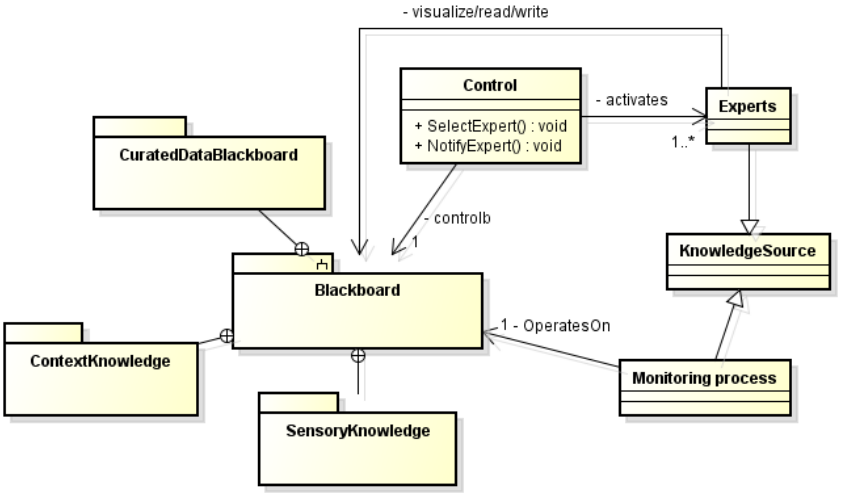

Figure 3.6: Cognitive monitoring management pattern for IoT and human interaction

The knowledge source can be the experts (human) who analyze the IoT data to extract new insights; or also the Monitoring process which receives the data from IoT devices and detects context changes. Based on the knowledge pattern, we decompose the blackboard component into SensoryKnowledge and ContextKnowledge, and we extend it with the CuratedDataBlackboard. The ContextKnowledge describes the target goals of the monitored data and time-related information of each sub-system, while the SensoryKnowledge describes the meaning of the generated IoT data, but not the observations. The CuratedDataBlackboard stores the prepared data for visualization and future analytics.

To deal with the heterogeneity of the data, the Monitoring process implements techniques that curate the data by referring to the SensoryKnowledge and store the prepared data in the CuratedDataBlackboard. Moreover, a smart IoT-based system should be able to keep the users up-to-date with the system context changes. Thus, the monitoring process refers to the ContextKnowledge to retrieve the system goal and to automatically detect problems. If detected, the monitoring process updates the ContextKnowledge; and the Control component selects and notifies the appropriated experts. It is worth mentioning that this pattern can be instantiated for data stream or batch processing.

Example. Using wearables and medical sensors to continuously monitor the prediabetes is promising to prevent the progress of prediabetes to diabetes [138] and avoid health complications. Figure 3.7 illustrated the application of the Cognitive Monitoring pattern for managing the patient health evolution.

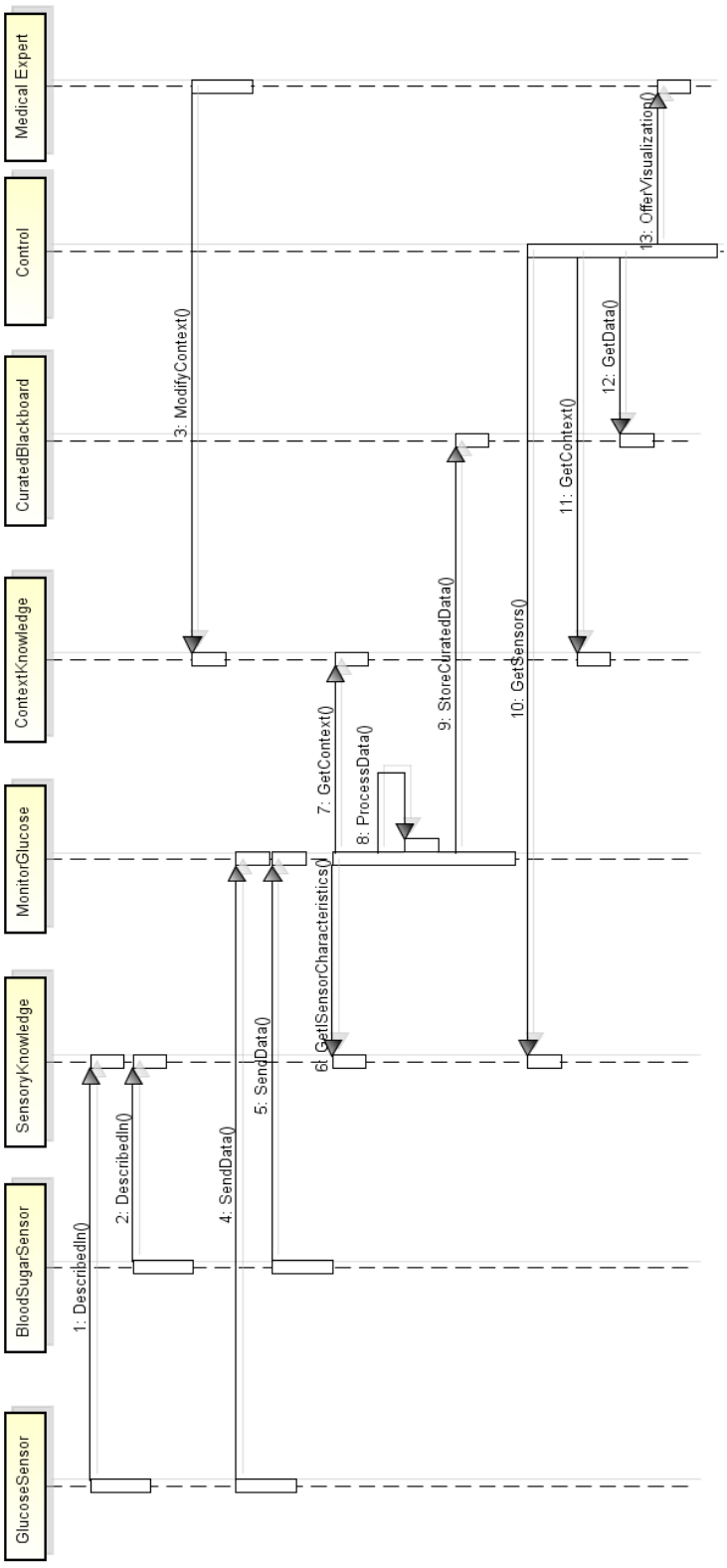

Figure 3.7: Instantiation of the cognitive monitoring pattern in healthcare

It allows continuously monitoring the glucose level and sending notifications to the appropriate physician at the right time when the measurement exceeds the patient target goal. Likewise, it allows the physician and the patient interacting with the systems to visualize the evolution of the monitored parameter. At this stage, the patient treatment management is manually done by the physician.

Consequence. Based on the sensory knowledge, the proposed pattern is able to manage heterogeneous IoT devices and offers a flexible system that can be easily extended with new devices that collect individualized data. The blackboard maintainability is guaranteed and can be done collaboratively. Problems related to the big data management will be delineated in section 3.4.4.

3.4.2.3 Predictive Cognitive Management Pattern

How the system is able to cope with the context evolution and detect unpredictable deterioration?

Context. Complex systems that continuously evolve need to implement advanced techniques that enable the dynamic coordination of the management processes in order to predict the system context changes, and generate new knowledge about the system to help experts taking the appropriate decisions.

Problem. Within real-world applications, unforeseen requirements may occur and need to be managed at run-time. If we consider deploying and activating all the monitoring and analysis processes at design time for each patient, this requires increasing the IT resources, consequently increasing the cost. Moreover, hard coding all rules within the analysis process causes maintainability and extensibility problems.

Solution. The proposed pattern is presented in Figure 3.8. We assume that all processes are deployed but activated based on the context changes. We propose to decompose the management processes into atomic functions and extend the cognitive monitoring management pattern to enable the coordination of three types of interactions:

- Monitoring-Analysis: The *ControlMA* supervises the *CuratedDataBlackboard*. If new data is acquired, it activates the appropriate Analysis processes to get more precise view about the managed element.
- Analysis-Analysis: The *ControlAA-AE* supervises the *ContextKnowledge*. If the context changes, new analysis processes should be activated based on rules implemented within the *ControlAA-AE*.
- Analysis-Expert: If a deterioration is detected/ predicted, the *ControlAA-AE* notifies the experts to take decisions.

The controllers encode rules that activate the appropriate process. Thus, new rules can be dynamically added and removed without modifying the code source of the management processes.

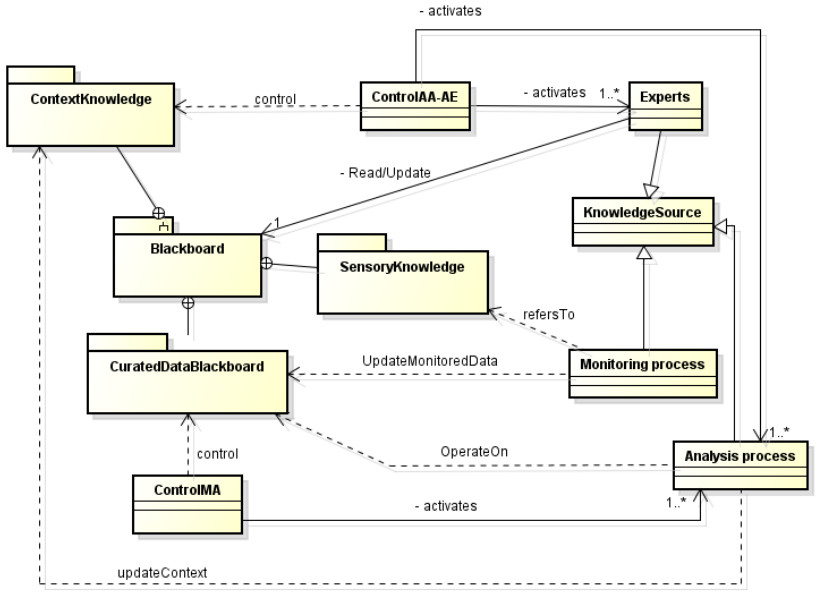

Figure 3.8: Predictive cognitive management pattern

Example. Figure 3.9 depicts an example of instantiation of this pattern for diabetes management. We consider that initially two management processes (*MonitoringGlucose*, *AnalzingGlucose*) are deployed to collect and analyze the glucose level of a patient who is recently diagnosed with diabetes. The patient is taking the metformin as a primary treatment. If a problem is detected, the *AnalzingGlucose* process updates the *ContextKnowledge*. Thus, the *ControlAA-AE* notifies the physician with the update in order to adjust the patient treatment. However, it is important to check the hypertension, each 3 months, to prevent health complications due to diabetes, or due to external factors such as stress, anxiety or aging. As this rule is implemented in the *ControlAA-AE*, the system activate the *PredictingHypertension* process that allows predicting the risk that the patient may develop hypertension based on other available parameters such the BMI, waist circumference, waist-hip ratio and waist-height ratio [139]. If the hypertension is predicted, a notification will be sent to the physician.

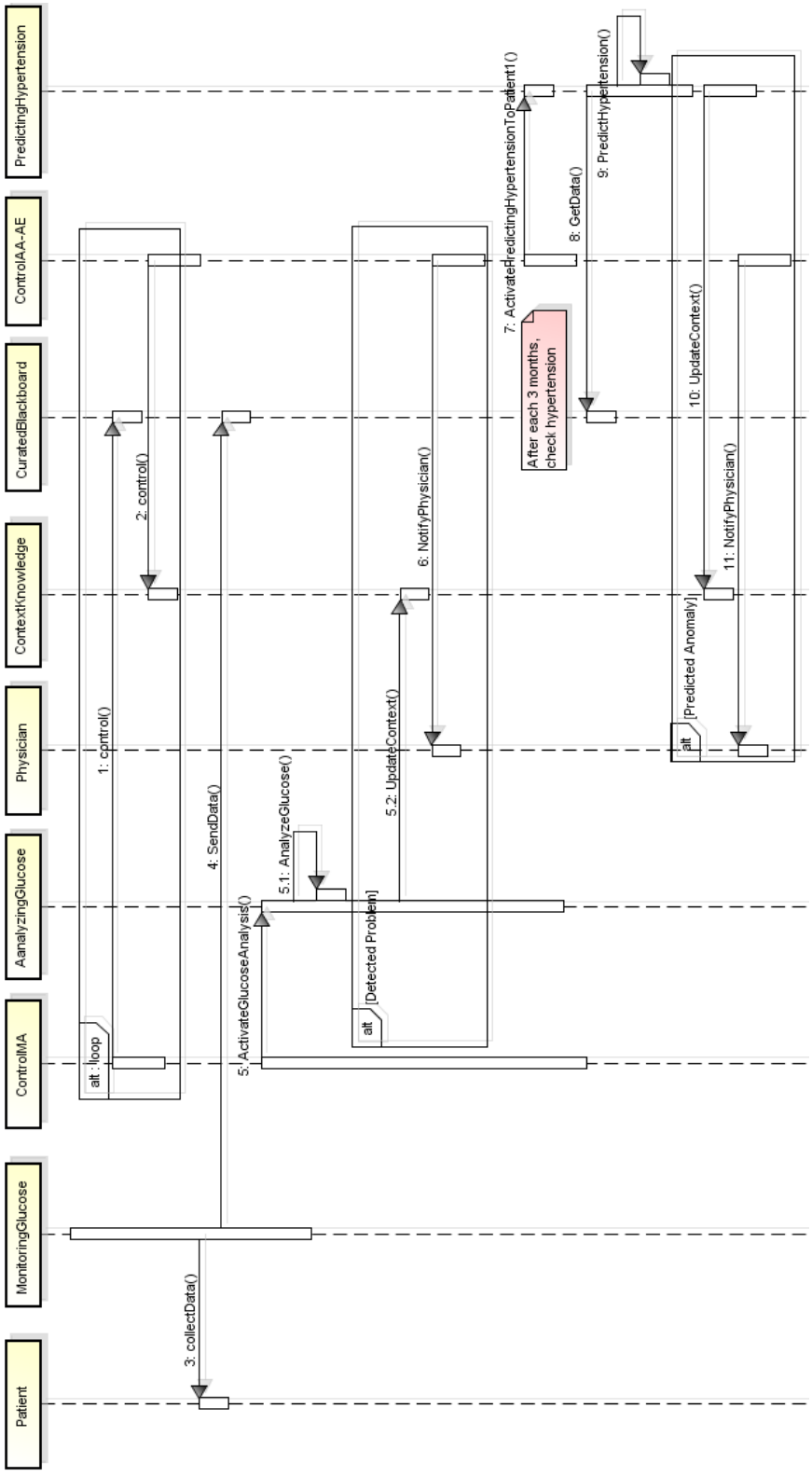

Figure 3.9: Instantiation of predictive pattern in healthcare

Consequence. The proposed pattern supports the dynamic coordination of the monitoring and the analysis processes, while interacting with experts based on the context changes. It also ensures extensibility and maintainability through modularizing the management processes, and separating the system management based on the control components. It enables the reusability of the management processes and the system flexibility to support new management processes and new coordination rules.

3.4.2.4 Prescriptive Cognitive Management Pattern

How the system is able to cope with context changes and generate personalized recommendations with minimum human intervention?

Context. Smart IoT-based systems should automatically monitor, analyze and provide decisions based on the context and the available knowledge. The system should be able to proactively generate new information about the managed element and to aggregate different knowledge sources in order to provide better decision. In business-oriented systems, the interaction with the business expert is mandatory for decision approval.

Problem. Due to the dynamicity of the sub-systems' context, a single MAPE loop is not sufficient for managing complex systems. We assume that initially the adaptation is based on enacting Plan (P1). If the context changes, the P1 generates the right adaptation and readjusts the current plan. However, the system context pertaining to other parameters may change and impact on the decision process. It is important to get updated/discovered information about the managed element in order to avoid contraindications and conflicts when re-planning. Thus, the coordination between the Analysis and Plan process is crucial to guarantee that these actions do not impact on other sub-systems. Moreover, the system requires a well-formalized procedural knowledge when reasoning to generate the right decisions.

Solution. We propose the prescriptive cognitive management pattern, presented in Figure 3.10, that extends the predictive cognitive management pattern through automating the Plan process and enable its interaction with the expert. Moreover, in order to allow the IoT-based system generating recommendations, we extend the blackboard with the *ProceduralKnowledge* describing the know-how for decision-making. The proposed pattern coordinates the following interactions: Monitoring-Analysis, Analysis-Analysis, Analysis-Plan, Plan-Expert. The generated plan will be stored in the ContextKnowledge; and it is the role of the *ControlPExpt* component to notify the appropriate expert for approval. The Plan process aggregates and reasons on the *ContextKnowledge* and the *ProceduralKnowledge* to provide personalized decisions.

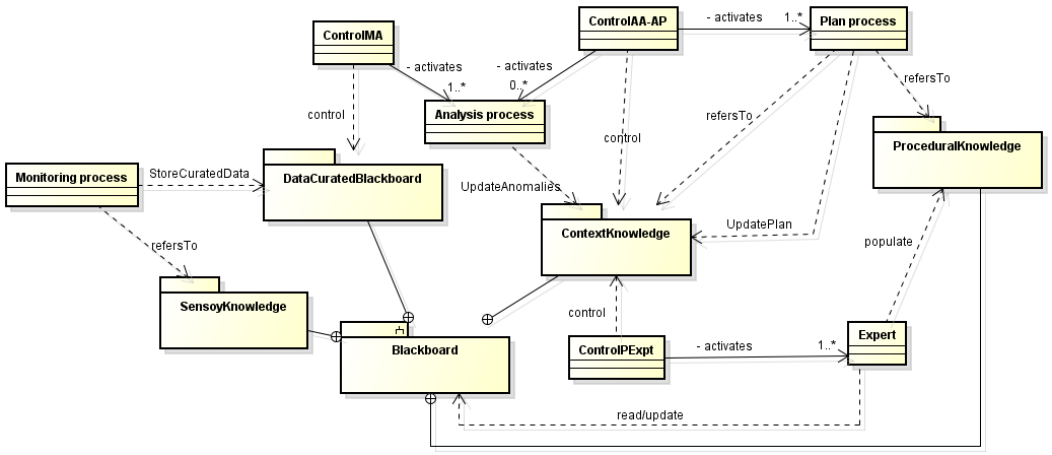

Figure 3.10: Prescriptive cognitive management pattern

Example. Figure 3.11 represents an instantiation of the Prescriptive Cognitive Management pattern for managing diabetes and preventing comorbidity. We assume that a patient has the following management processes for managing diabetes (*MonitoringGlucose*, *AnalyzingGlucose*, *PlanDiabetesTreatment*). Having a complete knowledge about the patient helps generating precise decisions. For instance, it is crucial to know if the patient has the risk to develop hypertension in order to avoid diabetes treatments that may amplify this condition. According to the Predictive Cognitive Management Pattern, analyzing the hypertension is triggered each 3 months. However, if the system detects an increase in the blood sugar after 2 months, it is important to check the hypertension when planning for the new patient treatment. Consequently, the *ControlAA-AP* activates the *PredictingHypertension* that updates the ContextKnowledge with new knowledge about the patient. Thus, the *PlanDiabetesTreatment* process may generate preventive treatment that avoid hypertension complication. The generated plan is updated in the *ContextKnowledge*,and the *ControlPExpt* sends it to the physician.

Consequence. The proposed pattern enables the dynamic coordination of the monitoring, the analysis, the plan and the expert to manage the system context changeability. It offers a smart management through automating the decision making based on the knowledge component, while the experts are in the loop to validate the recommendations. In the predictive cognitive and prescriptive cognitive management patterns, the management processes are deployed and known at design time, and their dynamic coordination is based on rules implemented in the control components. Thus, adding new management processes requires maintaining the code of the control components by adding the appropriate rules that support their activation.

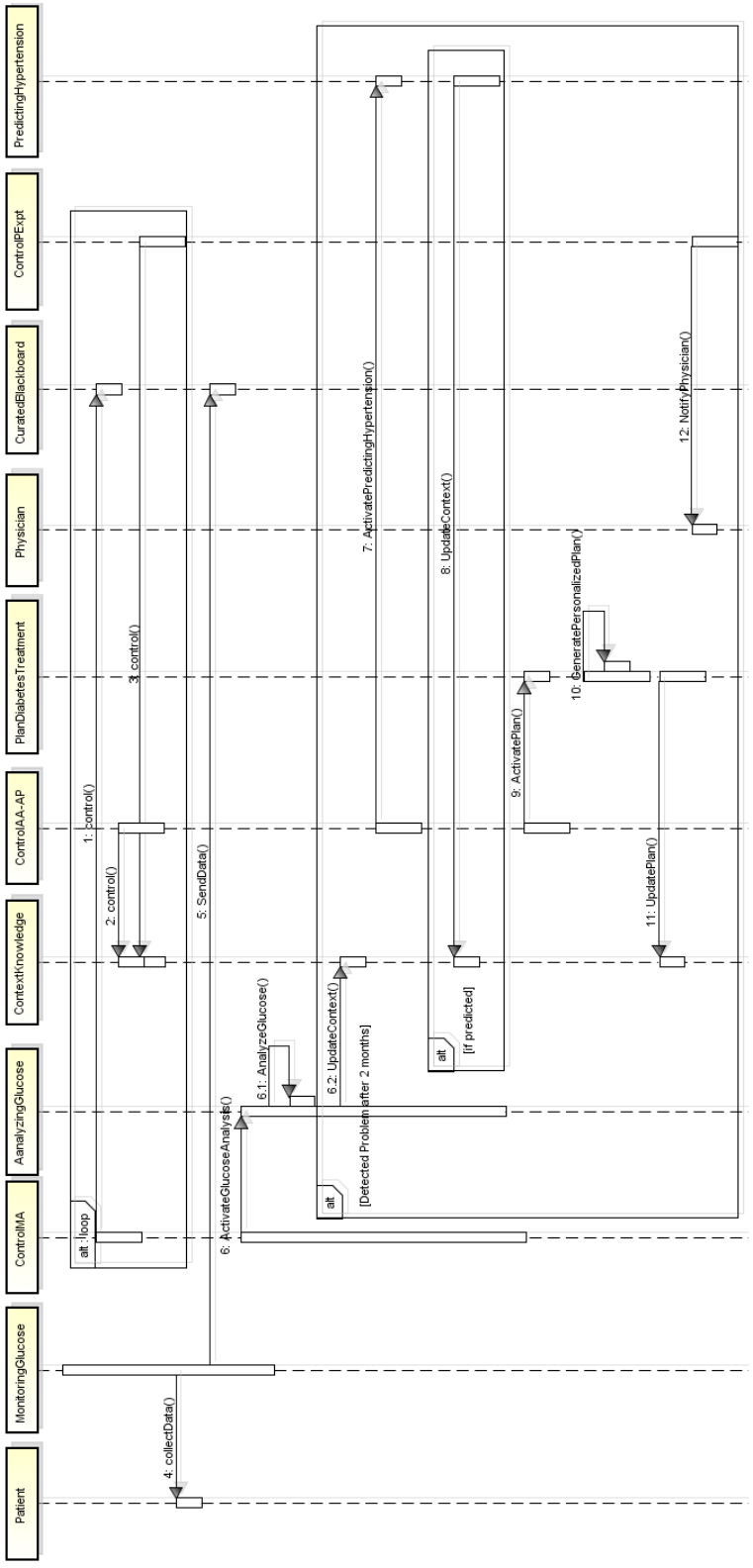

Figure 3.11: Instantiation of the Prescriptive Cognitive Management Pattern in Healthcare

3.4.2.5 Autonomic Cognitive Management Pattern

How the system is able to self-provision management processes in order to manage the context changeability?

Context. An autonomic system is a smart system that should be able to manage its changes and evolution at runtime. The dynamic evolution of the business context requires dynamically integrating the management processes implementing business logic, while interacting with the experts. Managing a complex system refers to simultaneously managing its sub-systems and coordinating the actions and their side effects that may impact on the system functioning and state evolution, while optimizing the system cost.

Problem. Management processes are heterogeneous and distributed, which hinder the integration and the collaboration of these processes to manage complex requirements at runtime. Moreover, in business-oriented applications such as healthcare, the interaction with experts is required to automate the execution of the generated plan as well as to learn business rules for the adaptation. Furthermore, in complex systems, the massive deployment of management processes may lead to an increased cost. It is important to think about the ability of sharing and reusing processes such as the analysis and the plan processes in order to help reducing the cost.

Solution. To deal with these problems, we propose to extend the prescriptive cognitive pattern with the Execution process and detail its interaction with the expert who validates the dynamic execution of the plan, as presented in Figure 3.12. Moreover, to ensure the self-provisioning of the management processes to meet the system's requirements evolution, we propose to extend the control components with a semantic model, named "Management Process Ontology" (MPO), describing the management processes as well as their conditions of activation, as presented in Figure 3.13. For clarity reasons, we simplified the representation of the previous patterns through abstracting the management processes and their interactions through introducing the "Management Process" class which represents a generalization of the Monitoring, Analysis, Plan and Execution processes. The coordination is achieved through the control components that automatically discover the management process that should be activated to meet the system context evolution.

Figure 3.13 delineates the *Management Process Ontology*. We associated for each managed element a set of conditions describing the context and a set of management processes. Each management process is considered as an atomic process that has a set of preconditions expressed as conditions in order to guarantee its activation and enactment. As previously mentioned, these management processes can be the "mpo:MonitoringProcess", the "mpo:AnalysisProcess", the "mpo:PlanProcess" or the "mpo:ExecutionProcess". As we consider all processes are atomic, so the "mpo:MonitoringProcess" monitors only one "mpo:Parameter"

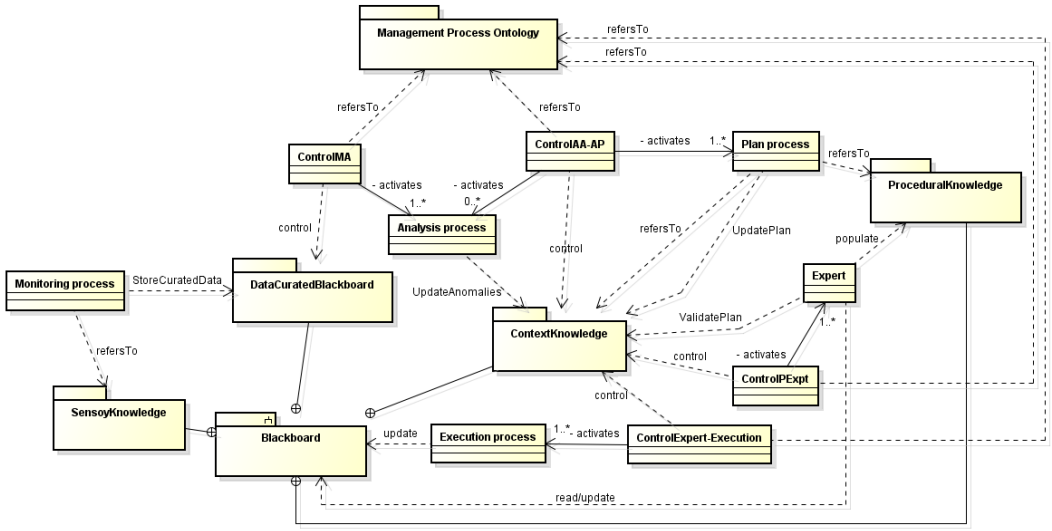

Figure 3.12: Autonomic cognitive management pattern

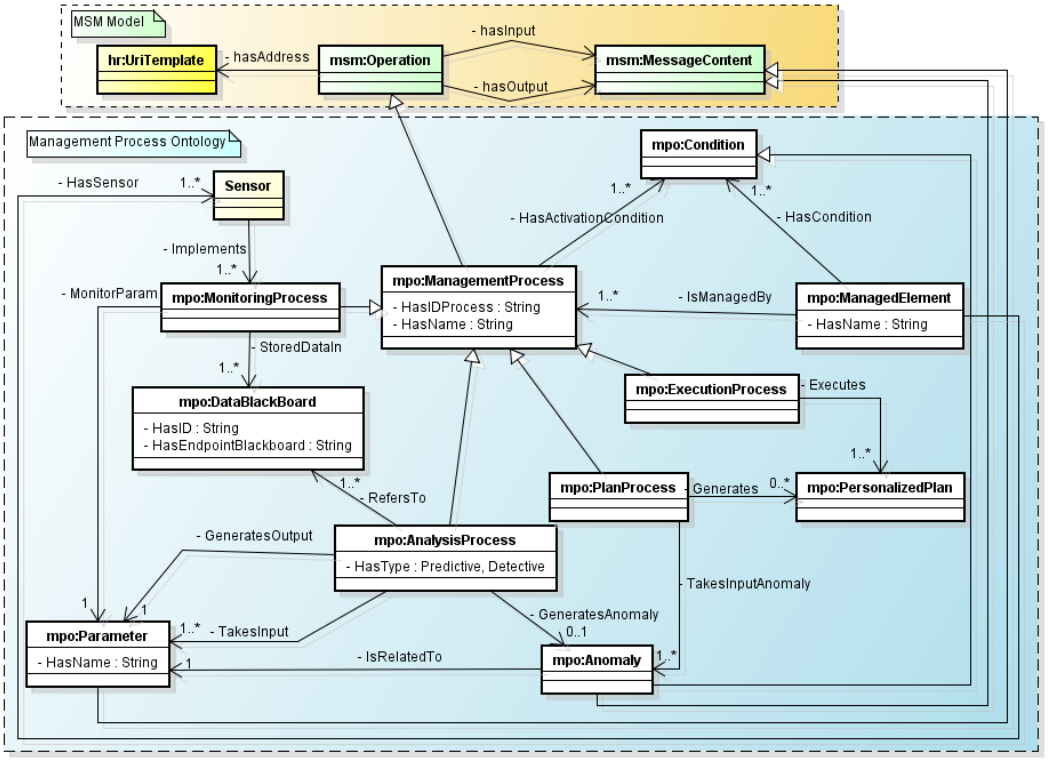

Figure 3.13: Management Process Ontology (MPO)

and stores the measured data in the "mpo:DataBlackBoard" identified through an ID and an endpoint that will be used to retrieve the stored data. In case of observing more than one parameter, the "mpo:ManagedElement" is supervised by multiple "mpo:MonitoringProcess".

MPO reuses existing ontologies such as the MSM ontology[3]. Thus, each management process is a sub-class of the "msm:Operation" that has inputs and outputs represented respectively through the "msm:hasInput" and "msm:hasOutput" properties. MPO specializes the "msm:hasInput" and "msm:hasOutput" by introducing the following properties to guarantee the consistency of the acquired knowledge:

- "mpo:TakesInput" to specify that the analysis process takes as input at least one parameter.
- "mpo:GeneratesOutput" to specify that the analysis process analyzes one parameter.
- "mpo:TakesInputAnomaly" to specify that the plan proces takes as input at least one anomaly.
- "mpo:Generates" to specify that the plan process generates a personalized plan.

An ontological representation of MPO using protégé is presented in Figure A.1. The MPO has been conceived to allow the business experts populating the business rules through creating relations among these classes. Thus, the system easily interprets the business logic and enables the self-provisioning through dynamically composing the management processes to meet the system evolution. To this end, a set of SPARQL queries are proposed to enable the dynamic discovery of the management processes and keep the control components up-to-date with new management processes. Listing 3.1 represents an example of a generic query that aims at discovering the monitoring processes and activating them based on the context changes. For instance, a patient with diabetes needs to check each 3 months her hypertension. Thus, the system should be able to search for possible available monitoring process to activate it. Thus, if the patient is equipped with a sensor that measures the blood pressure, it will activate this process, else, if the patient doesn't have a sensor, the control component executes the query presented in listing 3.2 to enable the predictive cognitive management of the hypertension.

Listing 3.1: Discovery of the monitoring process to activate based on the context changes

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
```

[3]http://kmi.github.io/iserve/latest/data-model.html

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX MPO:<http://homepages.laas.fr/emezghan/untitled-ontology
   -77#>
SELECT distinct (?el AS ?ManagedElement) (?s AS ?Sensor) (?ap AS
   ?MonitoringProcess ) (?endp AS ?Endpoint ) Where
{?ap MPO:HasEndpoint ?endp.
        {
        ?ap1 rdf:type MPO:MonitoringProcess.
        ?el MPO:IsManagedBy ?ap1.
        ?el MPO:HasSensor ?s.
        ?s MPO:Implements ?ap.
        Filter (?ap1 != ?ap).
    }
    {Select ?e ?ap Where
        {
            ?ap rdf:type MPO:MonitoringProcess.
            ?el MPO:HasCondition ?cond.
            ?ap MPO:HasActivationCondition ?cond.
         }
    }} GROUP BY ?ap ?el ?s ?endp
```

Listing 3.2 implements the query that allows discovering the required predictive analysis processes that should be deployed in order to provide preventive management based on the context of the managed element. This query is a sub-query, having three nested queries. At a first stage, it selects the list of management processes that have common preconditions with the managed element context. Then, from the list of the returned processes, it filters those all preconditions are satisfied and should be activated. The next step will focus on selecting from the generated list the processes that can be enacted based on the availability of the patient monitored data. For example, if the selected analysis process takes as input two parameters while the managed element has only one monitored parameter, the process cannot be activated due to knowledge/data incompleteness.

Listing 3.2: Discovery of the predictive analysis process that should be activated

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX MPO:<http://homepages.laas.fr/emezghan/untitled-ontology
   -77#>
Select (?el AS ?ManagedElement) (?ap AS ?AnalysisProcess ) (?endp
    AS ?Endpoint ) WHERE
{       Filter (?m1 = ?m2).
        ?ap MPO:HasEndpoint ?endp.
  {Select distinct ?el ?ap ?m1 (count(?p) AS ?m2) where
        {?ap MPO:TakesInputs ?p.
        {SELECT distinct ?el ?ap (count(?param) AS ?m1)  Where
```

```
          {
           ?ap MPO:HasEndpoint ?endp.
           ?ap1 rdf:type MPO:MonitoringProcess.
           ?el MPO:IsManagedBy ?ap1.
           ?ap1 MPO:MonitorParam ?param.
           ?ap MPO:TakesInputs ?param.
           ?ap MPO:HasType ?type.
           ?ap MPO:generatesOutput ?paramO.
           ?el MPO:IsManagedBy ?ape.
           ?ape MPO:generatesOutput ?param1.
           ?ap MPO:HasType ?type.
           Filter (?ap != ?ape).
           Filter (?param1 != ?paramO).
           Filter (?type = 'Predictive').
          } GROUP BY ?el ?ap ?m1
    }
        } Group by ?el ?ap ?m1 ?m2
  }
{Select distinct ?ap WHERE
{Filter (?c1 = ?c2).
{SELECT distinct ?ap  (count(?cond) AS ?c1)  (count(?cond1) AS ?
   c2) Where
 {
  {
        ?ap rdf:type MPO:AnalysisProcess.
        ?ap MPO:HasActivationCondition ?cond.
        ?el MPO:HasCondition ?cond.
          }
          UNION
          {
        ?ap MPO:HasActivationCondition ?cond1.
          }
 } GROUP BY ?ap ?c1 ?c2
}}}}
```

Example. We consider in this example managing patients with diabetes and hypertension diseases (comorbidity). A set of management processes for managing diabetes (*MonitorGlucose*, *AnalyzeGlucose*, *PlanDiabetes* and Execution) and hypertension (*AnalyzeHypertension* and *PlanHypertension*) are deployed and annotated in the MPO. Figure 3.14 describes the application of the autonomic cognitive management pattern. For clarity reasons, we abstract the blackboard and the control components. It is worth noting that the interaction among the management processes is done through the control components. As presented in Figure 3.14, Patient 1, who has diabetes, has a wearable that can measure blood sugar and blood pressure through two different interfaces implementing two measurement services. Initially, only the monitoring of the blood sugar is activated to continuously monitor the glucose level.

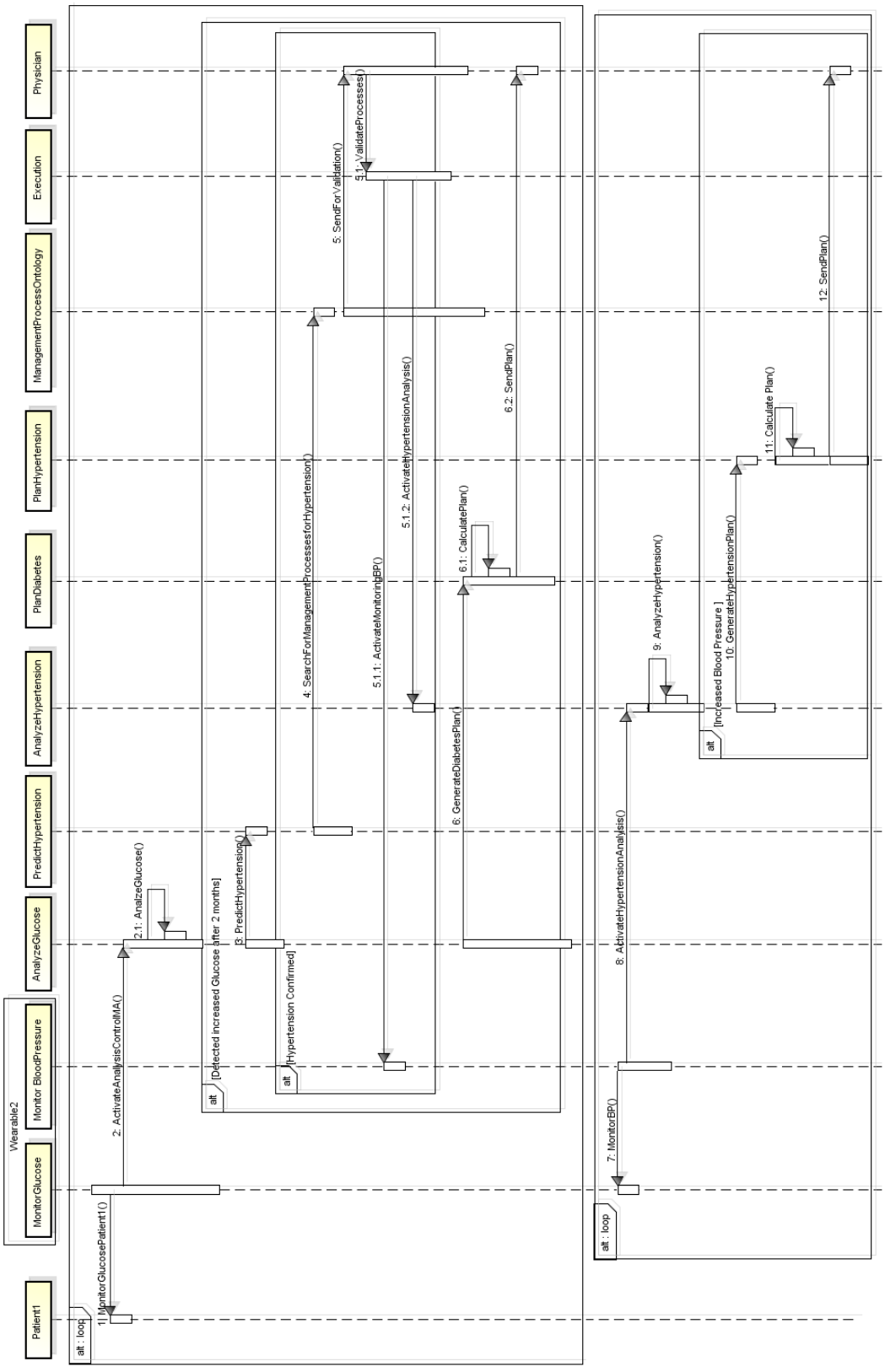

Figure 3.14: Instantiation of the autonomic cognitive management pattern in healthcare

If an increase of the blood sugar of Patient1 is detected after 2 months, the *PredictHypertension* process is activated. If the hypertension is predicted, the system searches for the list of management processes to control hypertension and sends it to the physician for validation. In this case, the management processes are *MonitorBloodPressure* and *AnalyzeHypertension*. Listing 3.1) represents a SPARQL query that deduces the activation of the *MonitorBloodPressure* based on the patient context, while Listing 3.3 represents the SPARQL query that searches for reusable analysis process once the monitoring process is deployed.

At the same time, the PlanDiabetes process takes into consideration the predicted hypertension, generates the appropriate treatment and sends it to the physician. At the end, the Patient 1 will be managed by both the diabetes and hypertension management processes.

Consequence. Reusing management processes such as the *Analysis* and the *Plan* process reduces the cost of deploying for each managed element its own management processes. Based on the semantic description of the management processes and the flexible implementation of the business rules, the proposed pattern enables the dynamic discovery and the self-provision of the management processes. The proposed pattern also keeps the experts in the loop for decisions approval.

Conflicts when generating plans may occur when two or more Plan processes managing dependent sub-systems are simultaneously operating. As a solution, a synchronization based on a token to manage the concurrency access to the ContextKnowledge may be deployed. However, we noted that when managing the patient treatment this situation is seldom encountered, especially that we consider a preventive approach based on the disease risk factors and patient medical conditions when generating treatments.

Listing 3.3: Discovery of the predictive analysis process that should be activated based on the new deployed monitoring process

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX MPO:<http://homepages.laas.fr/emezghan/untitled-ontology
    -77#>
Select (?el AS ?ManagedElement) (?a AS ?AnalysisProcess) (?endp
    AS ?Endpoint) WHERE {
   Filter (?c1= ?c2).
   ?a MPO:HasEndpoint ?endp.
   {
Select distinct ?el ?a (count(?param2) AS ?c1) (count(?param3) AS
     ?c2) WHERE
{
  {
   {
```

```
        {?m2 rdf:type MPO:MonitoringProcess.
         ?el MPO:IsManagedBy ?m2.
         ?m2 MPO:MonitorParam ?param2.
         ?a MPO:TakesInputs ?param2.
    }
        Union
        {?a MPO:TakesInputs ?param3. }
   }
  }
  {SELECT distinct ?el  ?a WHERE
   {
        ?el MPO:HasSensor ?s.
        ?m rdf:type MPO:MonitoringProcess.
        ?s MPO:Implements ?m.
        ?el MPO:IsManagedBy ?m.
        ?m MPO:MonitorParam ?param.
        ?a rdf:type MPO:AnalysisProcess.
        ?a MPO:HasType ?type.
        ?a MPO:generatesOutput ?param.
        ?a1 rdf:type MPO:AnalysisProcess.
        ?el MPO:IsManagedBy ?a1.
        Filter (?a != ?a1).
        Filter (?type = 'Detection').
   }
  }
} Group By ?el ?a ?c1 ?c2}}
```

3.4.3 Semantic Knowledge Mediator Pattern

How the system is able to understand the meaning of the IoT generated data and interpret the procedural knowledge in order to be reused by the management processes?

Context. Many existing distributed and heterogeneous data/knowledge sources exist with different representations and types. These sources can be sensors, applications, connectors to databases, evenly human. Enabling a smart management of the system requires the collaboration of these sources to get a deep understanding of the business knowledge (know-how) and detailed information about the system and its environment as well as their integration.

Problem. The heterogeneity of the knowledge representation hinders understanding the meaning of data, exploring and visualizing it. Moreover, if a new source is added to the system, modifications in its code source are required to take into consideration the new data source which is identified as impediments of the system maintainability and extensibility.

Solution. We propose in Figure 3.15 the *Semantic Knowledge Mediator* pattern that extends the basic Mediator pattern with semantic capabilities for data/knowl-

edge integration. The proposed pattern enables the collaboration of different kind of providers when populating the knowledge: human, machine and management processes. As presented in section 3.4.2.1, three type of knowledge are produced following different semantic models describing a common vocabulary of the domain. The annotations are stored in a large scale triple store. The access to the mediator to produce and consume the knowledge is driven by an authentication service.

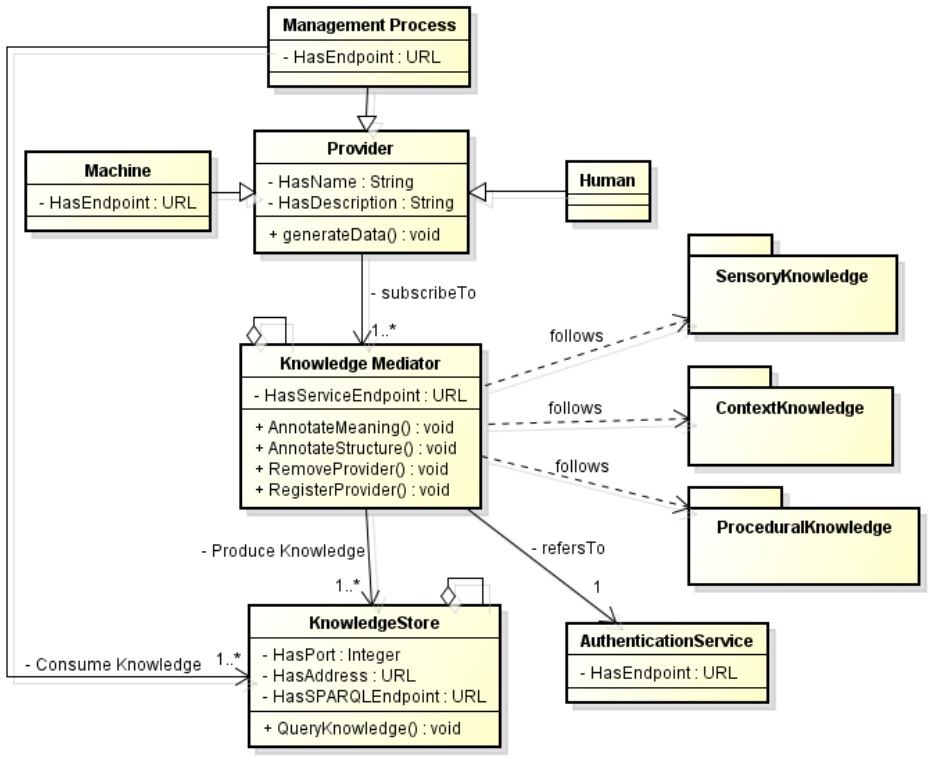

Figure 3.15: Semantic knowledge mediator pattern

Example. Instantiating the prescriptive cognitive management pattern for the patient treatment management requires the integration of various type of information related to the patient and to the medical interventions, as well as the collaboration of:

- The medical experts who transfer their medical knowledge through annotating the medical interventions (procedural knowledge).
- The management processes which generate new context about the patient and annotate it (context knowledge).
- Sensors, wearables and devices which are annotated in order to provide the meaning of the generated data as well as their structure (sensory knowledge).

Consequence. The proposed pattern enables the collaboration and offers the semantic integration of various knowledge sources based on a common description of the domain. It guarantees also the flexibility of the system and its extensibility to support new knowledge sources. Many research activities [36, 140] proposed to store the monitored data in ontologies. However, this approach raises scalability problems in such data-intensive systems. Consequently, we propose to characterize only the data structure and meaning and store these annotations in the ontology.

One major challenge comes out within this pattern is related to maintaining the knowledge schema that we have mentioned in section 3.4.2.1. But, as the knowledge is distributed and interconnected via common concepts, modifying some of these concepts requires updating the other knowledge concepts.

3.4.4 Big Data & Scalability Management Patterns

The monitoring and the analysis are the main processes which encounter big data challenges. Consequently, we propose the *Big Data Stream Detection* and the *Big Data Predictive* patterns as extensions of the Cognitive Monitoring Management pattern and Predictive Cognitive Management pattern in order to respectively manage the real-time processing, and the integration of distributed data sources to predict new information. Besides these patterns, we propose the *Multi-tenant Management Process* pattern which delineates the deployment of the management processes and the knowledge sub-components to provide scalable IoT-based system.

3.4.4.1 Big Data Stream Detection Pattern

How the monitoring process is able to perceive the received data at real-time while supporting its heterogeneity and high volume?

Context. Many vendors are developing their devices/sensors implementing their own applications which results in a diversity of the generated data. Offering a service to a wide range of patients to monitor the evolution of their health status as well as providing personalized detection requires guaranteeing the scalability to support huge data streams and the ability to correctly interpret the data.

Problem. Besides the heterogeneity, the data velocity and their huge volume hinder data integration for near-real time anomalies detection and data stream visualization. Manually harmonizing and curating IoT-generated data requires IT skills, which is costly and time consuming. Automating the real-time data processing is challenging within IoT-based systems due to the dynamicity of the generated data. Thus, it is important to provide a flexible and scalable IoT-based system that supports the integration of new IoT devices, and interacts with human through providing the right information at the right time.

Solution. We propose the *Big Data Stream Detection* pattern which represents a specialization for data stream processing and extends the *Cognitive Monitoring Management* pattern to cover big data and scalability challenges. Thus, we propose to decompose the *Monitoring* process into sub-processes that understand and curate the data based on the *SensoryKnowledge*, store the curated data in the *CuratedDataBlackboard* for visualization and analytic, and detect changes based on the *ContextKnowledge*. The ultimate goal of this pattern is providing a flexible IoT-based system able to perceive the data, retain attentions, and interact with experts. Figure 3.16 portrays the conceptual model of the *Big Data Stream Detection* pattern.

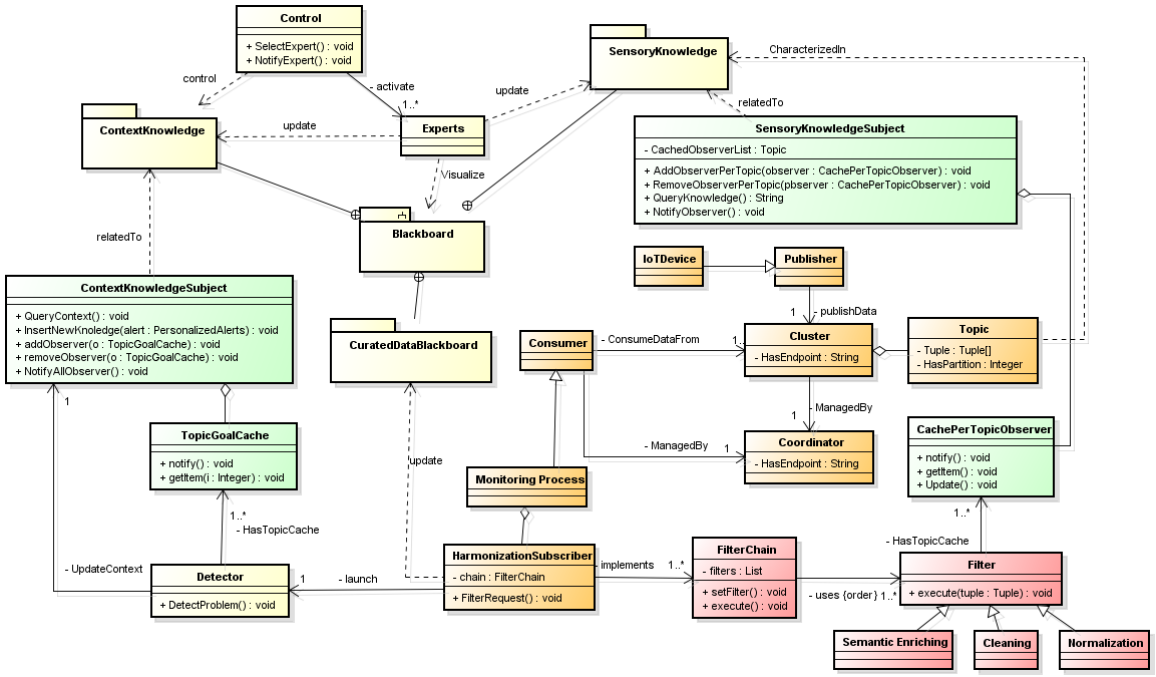

Figure 3.16: Big data stream detection pattern

In order to ensure flexibility when modeling these sub-processes and their interaction with the blackboard, we reused the following existing software design patterns:

1. *Publish/Subscribe*: it is represented with the orange color. It is used to guarantee the scalability between the data publishers (IoT devices) and the consumers responsible for harmonizing and/or detecting context changes. We added a coordinator in order to enable the distributed/parallel execution of tasks within consumers.

2. *Interceptor*: it is represented with the pink color. It is used to intercept the received data from topics in order to process data and harmonize it based on filters. We extended this pattern with a cache that retrieves semantic information concerning the IoT devices in order to provide flexible filters independent from the tuples structure.
3. *Cache*: it is used to guarantee system performance when retrieving semantic information from the *SensoryKnowledge* or the *ContextKnowledge*.
4. *Observer*: it is represented with the green color. It is used to keep the caches up-to-date, when changes occur in the *SensoryKnowledge* and the *ContextKnowledge* repositories.

In this pattern, we delineate the structure of the monitoring process to support the data heterogeneity and the scalability when receiving large amount of data. We introduced the Coordinator component in order to ensure fault-tolerance, since each piece of data may encapsulate important information that may change the system behaviour.

Example. For diabetes management, we can found sensors measuring blood sugar which express the observations in mmol/L and others in mg/dL. If a diabetic patient starts using a device generating data in mmol/L then switches to another sensor generating data in mg/dL, it is important to harmonize the generated data to offer near-real time visualization services and also to explore this data with advanced analytic tools. The application of this pattern enable such objective based on the semantic-based harmonization process.

Consequence. Based on the semantic description of the generated data, the proposed pattern fosters the stream data integration, and proposes a flexible harmonization process that can be shared and reused by various data providers. Likewise, it guarantees the system maintainability and flexibility through separating the data processing algorithms from the data structure and type. It offers also a scalable solution based on the publish/subscribe pattern and provides better performance through caching the sensory and context knowledge.

3.4.4.2 Big Data Analytic Predictive Pattern

How the monitoring and analysis processes are able to deal with the distribution and the heterogeneity of large datasets in order to generate new knowledge about the managed element?

Context. In data intensive systems, huge volume of heterogeneous data are being generated and stored in distributed databases. These databases should be integrated in order to learn from different data samples and generate new knowledge about the system. Providing a preventive system requires deploying predictive mechanisms that automatically correlate the collected data to detect new patterns.

Problem. Technically, the heterogeneity and distribution of databases hinder data mining and machine learning. Applying machine learning on a centralized large database requires both space to store the data, and memory to run the algorithm. Thus, it is a costly process in terms of time and money, especially for real-time applications where information should be timely available.

Solution. The proposed pattern, presented in Figure 3.17, extends the *Predictive Cognitive Management* pattern to support cognitive mechanisms to portray the aggregation of distributed large datasets. The first step focuses on curating the data to manage its heterogeneity. This process consists in importing data from external databases (if we have access) into a temporary memory, curating the data based on the *SensoryKnowledge*, and storing the curated data in distributed clusters (the long term memory –CuratedDataBlackboard) to be reused by machine learning algorithms. To optimize the memory space, the temporary memory is periodically cleaned.

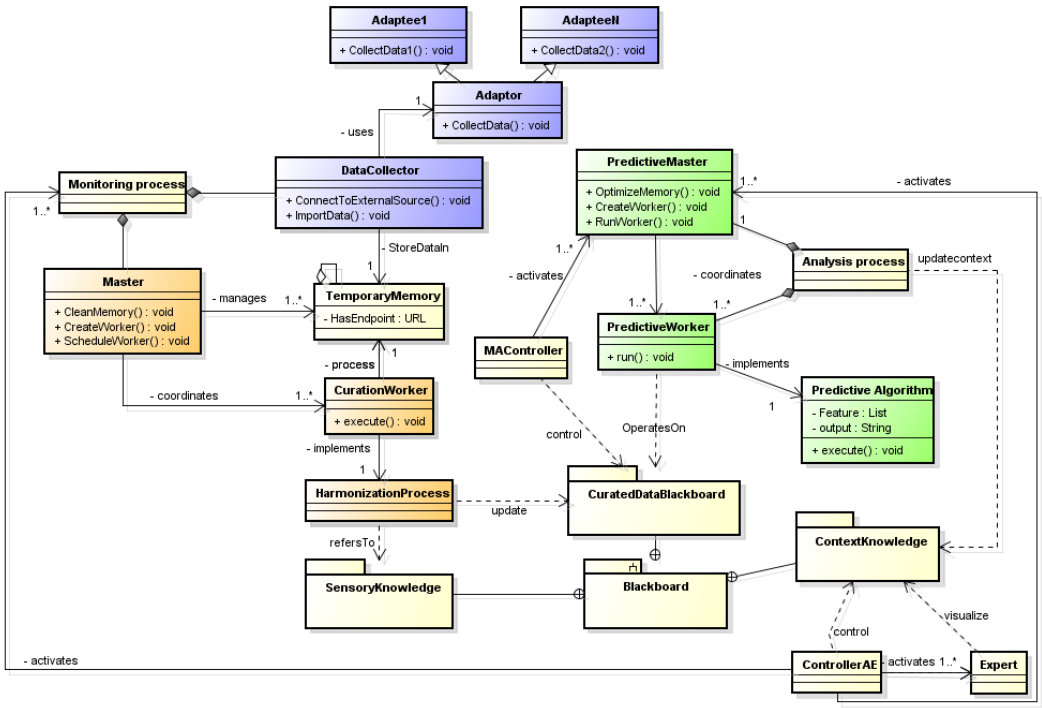

Figure 3.17: Big data analytic predictive pattern

The proposed pattern adopts the following existing patterns:

1. *Adapter*: It allows reusing existing databases without modifying their implementation.

2. *Filter Chain Responsibility*: Similar to the *Big Data Stream Detection* pattern, the harmonization process implements filters that refer to the *Sensory-Knowledge* describing the characteristics of the databases.
3. *Master/Slave*: It has been used twice: (i) in the harmonization process where we introduced CurationWorkers which are executed in parallel on the temporary memory to accelerate this process, and (ii) in the data analytic where we identified PredictiveWorkers which read part of the data and execute the analytic algorithm. This later can be easily implemented with the Apache Spark MLlib library.

Example. In case of comorbidity, the patient has more than one disease such as diabetes and hypertension. Each disease is managed by the appropriate physician who belongs to a specific healthcare organization implementing its own database. Thus, the patient data is stored in different databases with different format (SQL, text file, etc.). To predict the hypertension based on data related to diabetes, it is important to aggregate both databases in order to learn from data pertaining to diabetic patient with hypertension and without hypertension, and construct models that allow predicting the risk that a diabetic patient may develop hypertension [141]. The application of this pattern allows aggregating these databases and providing analytic services.

Consequence. Similar to the *Big Data Stream Detection* pattern, the semantic description of the external databases structure fosters the data integration as well as the system maintainability. Additionally, the adapter pattern allows easily integrating new external databases in order to enrich the database with new data, while the Master/Slave pattern is used for data ingestion and analysis. This later ensures fault-tolerance and parallelism when processing data.

3.4.4.3 Multi-tenant Management Process Pattern

How to manage the cost of autonomic and cognitive IoT-based systems as well as its performance and confidentiality?

Context. Deploying multiple management processes, knowledge components and databases increases the system complexity and cost. Moreover, when conceiving IoT-based systems, it is important to consider the confidentiality of the monitored data, especially in business-oriented application.

Problem. In complex System of Systems, different sub-systems need to be managed. The deployment of the management processes, the knowledge components and databases for each managed element will increase the system complexity as well as the cost in terms of memory, network traffic and CPU consumption. Consequently, the cost of the offered services in terms of money (customer side) increases in turns. Moreover, data should not be mixed.

Solution. The proposed pattern, presented in Figure 3.18, is the combination of (1) the cloud computing virtualization that allows reducing the system cost through sharing the resources among the management processes, the knowledge stores and the databases; and (2) the multi-tenancy that allows multiple users sharing the same process instance while isolating the context knowledge and the database of each tenant to guarantee confidentiality.

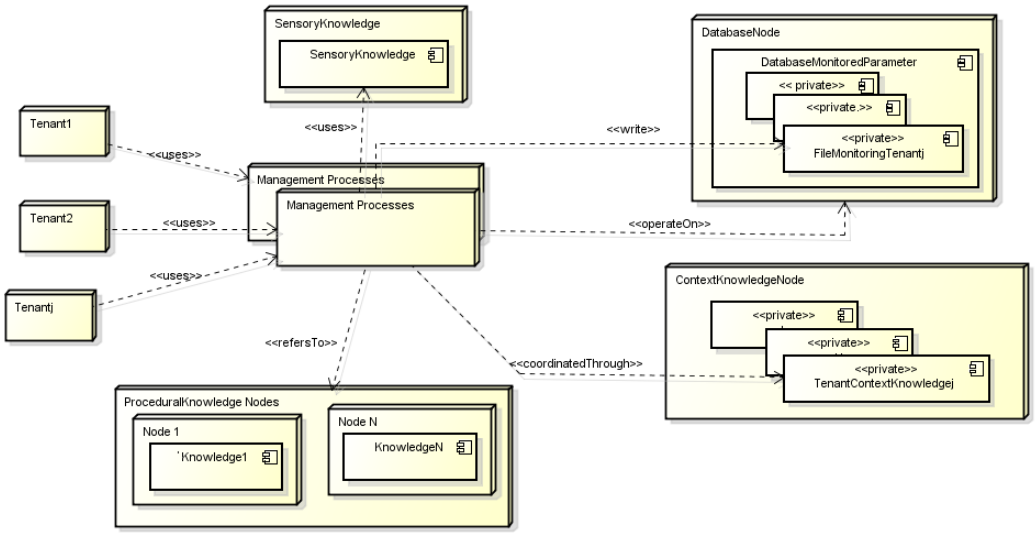

Figure 3.18: Multi-tenant management process pattern (UML deployment diagram)

For clarity reasons, we abstract the representation of the management processes interactions as they have been delineated in section 3.4.2. The tenants (consumers) may share the same processes with customized configuration or not, depending on the system requirements. During the customization, access control policies should be identified to control the access to the database and to the context knowledge.

Example. If we are managing complex systems such as a high number of patients with diabetes, deploying for each sub-systems (patient) its own processes requires allocating more resources (increasing the cost). Based on virtualization, it is possible to share and reuse the same IT resources among various processes such as the *Monitoring*, the *Analysis* and the *Plan* processes.

Consequence. Based on the cloud computing virtualization, the proposed pattern is able to reduce the cost through sharing the resources among the processes, the knowledge and the database. Moreover, it allows enhancing the system performance through horizontally scaling by adding new resources. Also, based on multi-tenancy, the system isolates the managed element' context and data in order to guarantee the confidentiality.

3.5 Conclusion

In this chapter, we delineated our collaborative model-driven methodology for designing autonomic and cognitive IoT-based systems. Within this methodology, we proposed a set of generic patterns classified as follows: four patterns describing the management processes coordination, one pattern enabling the semantic integration to generate three types of knowledge (sensory, context and procedural), two patterns dealing with big data challenges and finally one pattern dealing with the scalability management. The definition of these patterns has been motivated by the need in healthcare domain. We illustrated the use of the proposed patterns based on concrete examples pertaining to managing the treatment of patients with chronic diseases, and we detailed the consequences of using them as well as their limitations.

In the next chapters, we propose instantiating the proposed patterns for patient health management. Thus, we provide a semantic cloud-based big data platform that allows integrating heterogeneous data sources based on the management processes in order to generate new knowledge offered as a service.

CHAPTER 4

A Knowledge as a Service Platform for Heterogeneous Wearable Data Integration

"A vast technology has been developed to prevent, reduce, or terminate exhausting labor and physical damage. It is now dedicated to the production of the most trivial conveniences and comfort."

- Burrhus Frederic Skinner

Contents

4.1 Introduction 82

4.2 Wearable Computing in Healthcare 82

4.3 Existing IoT Platforms 84

4.4 Knowledge as a Service for Semantic Cloud-based Big Data Management 88

4.4.1 Overview of the KaaS Architecture 88

4.4.2 Wearable Healthcare Ontology 90

4.4.3 A Cognitive Monitoring System for Patient Health Management 93

4.5 Performance Evaluation 97

4.5.1 On-premises vs Cloud Evaluation 97

4.5.2 Multi-Node Cloud Evaluation 102

4.6 Conclusion 106

4.1 Introduction

By enabling communication and data exchange amongst heterogeneous devices, the IoT ultimately involves huge amount of data which analytics can offer new opportunities for business development and/or accurate decision making. However, its integration unquestionably points out challenges related to the heterogeneity of the data, to the system scalability, and performance.

Taking the wearable technology as a clue success of IoT, in this chapter, we highlight its trends in healthcare as well as the challenges derived from their adoption. Then, we discuss existing IoT platforms dealing with the heterogeneity and big data management. After that, we detail our proposed *Knowledge as a Service (KaaS)* platform for the integration of heterogeneous data source to generate knowledge about the system. Based on the patterns proposed in chapter 3, we provide a cognitive monitoring system combining the semantic web, big data and cloud computing technologies to create unprecedented opportunities that allow timely processing and analyzing health big data stream stemming from heterogeneous wearable devices.

As an illustrative example, we refer to the healthcare as applicative domain. Thus, we instantiate it for managing the patient health evolution. To this end, we propose the *Wearable Healthcare Ontology* (WH_O) which aims at providing a common description of the wearable data. This ontology is the foremost element that allows the system dynamically understanding the meaning of the received data. Finally, we evaluate the system performance within the KaaS, in terms of response time and scalability management as well as evaluate the associated function cost.

4.2 Wearable Computing in Healthcare

The wide adoption of wearable devices propels industries and researchers to team up together and provide more efficient solutions to track the human activities and continuously monitoring the patient's vital signs [142, 143, 144, 145]. Their integration in healthcare to monitor patients with serious conditions contributes to potentially reducing the healthcare cost by 88%[1].

Recently, Penders et al. [146] have pointed out the importance of tracking the lifestyle behaviors including physical activity, sleep, stress, diet, and weight management based on wearable sensors during pregnancy. The objective is to adapt and personalize the life style behaviors based on the collected data to provide healthier pregnancy. The use of wearable devices also pinpoints its impetus in the Active and Assisted Living (AAL) area, which aims at helping the disabled persons and elderly

[1] http://healthcare.orange.com/eng/news/latests-news/2014/infographic-wearable-tech-boom-in-healthcare

to offer a better quality of life. For instance, the work of Nicolelis [147] proposed to connect the brain to external devices in order to transform the brain signals into actions executed by the machine, such as moving the limbs just by thinking [13], to help people suffering from catastrophic body paralysis performing the desired action[2].

Nowadays, the wearable market focuses on producing a new range of tiny wearables embedded within clothing and accessories to provide more efficient services and offer an easy interaction with. For instance, Ford is collaborating with RWTH Aachen University to integrate heart-monitoring sensors in the car seats to detect abnormal heartbeat and heart attacks. If detected, automated steering and braking systems will be activated[3]. Furthermore, Google X research lab has collaborated with the pharmaceutical business Novartis and Alcon's to create smart contact lenses[4] that measure glucose levels in tears for diabetes patients and correct vision for people with presbyopia. At the University of Southern in Los Angeles, computer scientists and medical experts collaborate together and created an algorithm that uses data generated by various sensors including body sensors to better treat Parkison's disease [148]. In this way, medical experts can evaluate the treatment efficiency and notify patients. Other research activities focus on managing IT challenges related to the integration of IoT. For instance, IBM Watson Health and Apple have announced a new collaboration that focuses on providing cloud-based platform for a secure management of the patient data. Based on Apple ResearchKit, IBM's secure cloud and advanced analytics capabilities provide additional tools to accelerate discoveries across a wide variety of health issues[5].

The wearable market witnesses an important increase. According to the International Data Corporation (IDC)[6], the worldwide wearables market forecast shipped 45.7 million units in 2015 and it is expected to reach 126.1 million units in 2019. Nevertheless, the rapid growth of this marker and its adoption create challenges in the development of smart IoT healthcare systems. The heterogeneity and the large amount of the generated data remain the main challenges from the data management and computational perspectives. In the next section, we discuss existing IoT platforms and approaches that have been proposed to semantically integrate data generated by the IoT/wearable systems.

[2]http://www.ctvnews.ca/sci-tech/cyborg-soccer-how-a-paraplegic-took-first-kick-at-the-world-cup-1.1868837

[3]http://www.medtees.com/content/ecg_seat_fact_sheet_2.pdf

[4]http://www.cnet.com/news/google-extends-smart-lens-tech-for-those-with-diabetes-vision-problems/

[5]IBM Press: http://www-03.ibm.com/press/us/en/pressrelease/46583.wss

[6]International Data Cooperation: http://www.idc.com/getdoc.jsp?containerId=prUS25519615

4.3 Existing IoT Platforms

Despite the development of standards, interoperability and data integration are still open issues. In healthcare, the work of Kim et al. [149] is an example of research efforts that deal with integrating the HL7 with the IEEE 1451 standard to ensure interoperability when monitoring the patient data. The heterogeneity of wearable devices impedes their integration into existing systems which require additional efforts to develop applications from scratch [150].

Ontologies and semantic description have been widely used to provide a common vocabulary and representation of sensors. The Semantic Sensor Network Ontology (SSN) [78], which is proposed by W3C, is the most famous ontology describing sensors and devices. Many works adopted SSN in defining IoT platforms in order to deal with the heterogeneity of the sensors. IoT-A [35] is an IoT reference architecture that has been proposed to enable the interoperability among IoT connected devices. It describes sensors properties based on SSN, the service model with OWL-S and extends them with an IoT information Model [151]. Moreover, IoT-A integrates the cloud computing for complex event processing [152] to guarantee scalability and efficiency.

OpenIoT [34] is another platform that uses X-GSN to annotate the sensors and observed value based on SSN, and stores them in RDF stores in a cloud infrastructure in order to guarantee the scalability and the elasticity of the platform. Based on the semantic annotation, OpenIoT enables the semantic search and discovery of sensors and services. The observed data is stored as linked data and processed based on SPARQL queries which are continuously executed once data arrive. Sensors and devices are connected to the X-GSN middleware via publish/subscribe mobile broker in order to guarantee near real-time management.

Recently, Ben Alaya et al. [37] enriched the SSN ontology with the description of actuators. The authors proposed the IoT Ontology (IoT-O) for the autonomic management of M2M systems. The IoT-O is the amalgamation of various existing ontologies such as SSN, DUL, HREST, ACT, TIME, MSM and QUDT. IoT-O describes sensors and actuators, as well as the observed values and the services offered by each device. It has been instantiated for the management of smart home.

In healthcare, Lasierra et al. [36] proposed an ontology to describe the patient's vital signs and to enable semantic interoperability when monitoring patients data by formalizing the X73 standard. Following the same direction, Kim et al. [38] proposed an ontology driven interactive healthcare with wearable sensors (OdIH_WS) to acquire context information at real-time using ontological methods by integrating external data such as meteorological web site in order to prevent disease. However, the work of [36] and [38] did not consider the system scalability.

The adoption of IoT emphasizes the big data phenomenon and raises scalability challenges related to the data storage as well as to the computational and near real-

time data processing. In this context, Xu et al. [153] proposed the Time Series analytics as a Service (TSaaaS), a cloud-based analytic service for time series data in IoT scenarios. The aim of TSaaaS is to provide scalable pattern search service that leverages the large amount of IoT generated data with low search latency and high search accuracy. TSaaaS is based on the IBM Cloud platform, stores IoT data in IBM Informix database server, and offers RESTful interfaces to simplify the search process. It offers a faster pattern search compared to existing pattern search techniques, with additional storage cost. However, the proposed solution does not detail how the heterogeneity of IoT data is managed.

Mingozzi et al. [154] proposed the Building the Environment for the Things as a Service (BETaaS) for the integration of distributed and heterogeneous existing IoT systems. The solution adopted in BETaaS concentrates on exposing things as services (TaaS) through service-oriented interfaces. Thus the integration is achieved with limited efforts and modifications. BETaaS is a semantic-driven solution where two ontologies [155] are defined: the BETaaS Things Ontology which reuses existing ontologies such as SSN, OWL-Time and QUDT, and the BETaaS Context Ontology which is the integration of the BOnSAI ontology, GeoNames ontology and GeoSPARQL ontology. BETaaS [122] includes also a big data manager in the TaaS and service layers that have the main functionalities gathering, storing, adapting, processing, and analyzing data. However, TSaaaS [153] and BETaaS [154] are not following a well-accepted cloud computing and big data reference models.

Sowe et al. [124] proposed a big data cloud platform for the management of data stemming from sensors. The proposed platform helps research scientists easily discovering and managing data from various sensors, as well as sharing their knowledge and experience relating to air pollution impacts. The authors identified the following cloud layers:

- The IaaS includes a set of virtual machines and the Service-Controlled Networking that collects and analyzes data from physical and social sensors.
- The PaaS includes distributed databases (MySQL, MongoDB, etc) named SDaaS and interact with the users based on MediaWiki to enable collaborative authoring and build a knowledge repository from a variety of sensors.
- The SaaS includes customized web-based applications that allow, for example, visualizing sensors data, tracking and discussing the quality of the data.

However, there is a lack of support for the diversity and heterogeneity of the received data in terms of units and format, especially that many IoT vendors are introduced in the market using various standards and data representation.

In healthcare, Forkan et al. [156] proposed a cloud-based context-aware system called CoCaMAAL which covers challenges related to data collection and data

processing in ambient assisted living systems. The authors proposed to mitigate the complexity of data computation from sensors to the cloud. They identified an abstract ontology to describe the context including patient information, the environment and devices. Jiang et al. [157] are interested in big data solutions for wearable systems in healthcare. They proposed a wearable sensor system with an intelligent information forwarder that adopts the Hidden Markov Model (HMM) to estimate the hidden wearer's behaviors from sensor readings, and to determine the probability that the patient has a specific health state. The heterogeneity of data units, format, and representation generated by different wearable devices remains challenging in the works of Forkan et al. [156] and Jiang et al. [157].

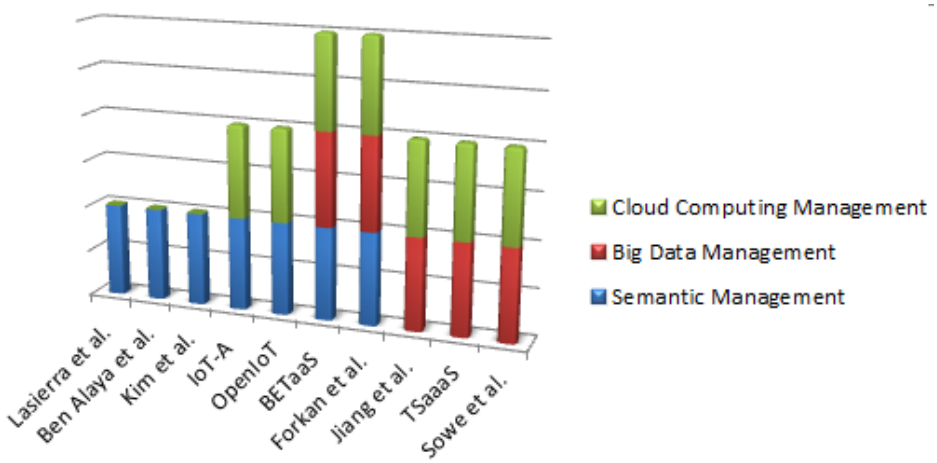

Figure 4.1: IoT Platforms' approaches

Figure 4.1 provides a visual classification of the discussed IoT platforms following three approaches: the semantic management to ensure interoperability and integrability; the big data management to provide accurate analytics while managing the data volume and velocity; and the cloud management to guarantee scalability and elasticity. Table 4.1 highlights the techniques and tools used by each work. We noted that few research activities focus on coupling these approaches. Almost works supporting the semantic management of IoT-based systems store the observed data in ontology which may cause scalability problems when reasoning. Moreover, it is important to guarantee the confidentiality and the privacy when storing the monitored data (patients' data should not be mixed in the same files/table/RDF Graph). Furthermore, none of the aforementioned platforms clearly demonstrate their flexibility and extensibility to support the integration of new data sources for better decisions.

Related Works	Semantic Management	Big Data Management	Cloud Computing Management
IoT-A [35]	SSN and OWL-S ontologies	-	Yes: Cloud CEP
OpenIoT [34]	SSN	-	Yes: Data Storage
Lasierra et al. [36]	Vital signs description X73 standard	-	-
Ben Alaya et al. [37]	IoT-O	-	-
Kim et al. [149]	OdIH_WS	-	-
Forkan et al. [156]	Patient Context, devices	Yes	Yes
Jiang et al. [157]	-	Yes	Yes
TSaaaS [153]	-	Yes: IBM Informix database server (SQL/NoSQL)	Yes: IBM BlueMix Cloud Platform
BETaaS [154]	Yes: Things ontology & Context ontology	Yes	Yes
Sowe et al. [124]	-	Yes	Yes: IaaS, PaaS/SDaaS and SaaS

Table 4.1: A survey of IoT platforms: approaches and techniques

Concequently, we propose the *Knowledge as a Service* platform that relies on ontologies to describe the characteristics of data providers, on cloud computing to lighten the computational tasks and expose the knowledge as service, and on big data to manage the large amount of data and its velocity. Compared to existing platform, our KaaS leverages cloud computing and big data technologies to store measurements in order to guarantee the system scalability, and uses ontology to tag wearable devices characteristics and data types to guarantee interoperability and integrability. Within the KaaS, analytic services and information extraction methods are exposed as services. They are flexible enough to be reused by different consumers, which is challenging in healthcare monitoring system [18].

4.4 Knowledge as a Service for Semantic Cloud-based Big Data Management

In this section, we present a generic semantic big data reference architecture adopting the Knowledge as a Service (KaaS) approach to deal with data heterogeneity and system scalability challenges. Then, we specialize it to healthcare for managing patient health.

4.4.1 Overview of the KaaS Architecture

Many research activities [130, 128, 129, 127] proposed the Knowledge as a Service (KaaS) to enable the collaboration of distributed data providers through generating knowledge from heterogeneous data and make it available as a service. However, none of these works built their solutions based on well-accepted cloud architecture or consider the big data management related to IoT-based systems.

From the cloud computing perspective, we proposed the KaaS layer as an enrichment of the NIST cloud computing architecture [118]. We introduced the KaaS as a new sub-layer on top of the standard PaaS layer and under the SaaS layer [158]. This definition has been motivated by the fact that the KaaS refers to the PaaS, where the collected data is stored in a distributed way, in order to extract new knowledge that will be consumed by the SaaS for better decision-making and visualization services. By means of the PaaS, the KaaS layer is able to use virtual and physical network resources available in the IaaS layer to integrate distributed and collaborative knowledge sources. From the big data perspective, our KaaS implements the NIST Big Data reference architecture [159] defined by the NIST Working Group, where 5 abstract components are identified: data collection, data curation, analytics, visualization and access control.

As multiple data sources are being available and following various data representation with different semantics, integrating a new existing data provider to the

KaaS is challenging and requires IT skills to maintain the system to consider the new received data. To this end, we propose enriching the NIST Big Data reference architecture with Semantic Web (1) to cope with the heterogeneity of data and (2) to transform the result of the analytics into sharable and reusable knowledge. Figure 4.2 depicts the proposed KaaS architecture. We identified three layers: the *Data Storage Layer* where large datasets are stored in the PaaS; the *Big Data Layer* that includes the NIST Big Data components associated to scalable technologies for data processing; and the *Semantic Knowledge Layer* that offers the common understanding of data. This architecture allows preparing data to be interpreted and reused by computers and human, and to be easily integrated with external information systems, independent of its structure and its representation.

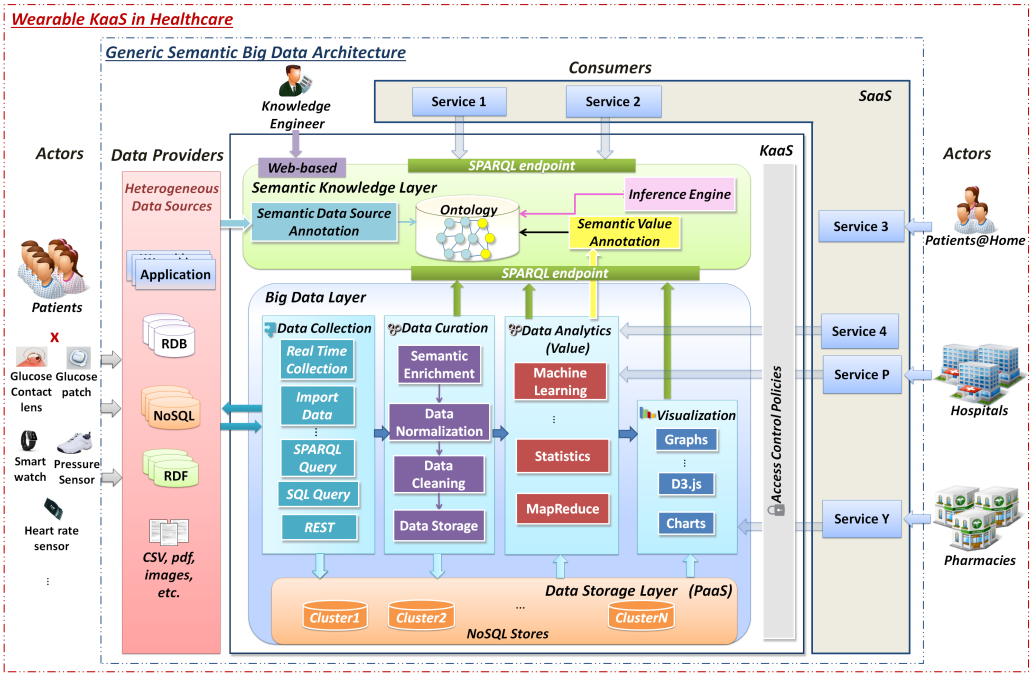

Figure 4.2: The generic conceptual KaaS architecture

Based on ontology, the *Semantic Knowledge Layer* formalizes the sensory knowledge (colored in blue) and the context knowledge (colored in yellow) introduced in the Knowledge pattern (see chapter3, section 3.4.2.1). The sensory knowledge, which is acquired through the "*Semantic Data Source Annotation*" service, describes the data providers' characteristics (e.g. the monitored parameter, its meaning, its storage location, etc.). It is used by data processing services to sense and perceive the meaning of the monitored data. The context knowledge, which is acquired through the "*Semantic Value Annotation*" service, describes the managed elements goals and status. It is used by the analytic services to detect/predicted

the context changes. The knowledge is stored in an RDF triple store and queried through SPARQL which is a standard recommended by the W3C.

Once the data providers are known by the KaaS, the data collection is triggered within the *Big Data Layer*. We identified two modes: the pull mode in which the data are collected at real-time from wearables and sensors, and the push mode in which the data is imported from external databases through using the appropriate API (SPARQL, SQL, etc.) depending on how the data is represented. The gathered data is stored in distributed big data clusters deployed in the *Data Storage Layer* (PaaS).

The next step focuses on harmonizing the collected data to be processed by the domain analytic services. Data curation operates on the semantic layer and the data storage layer in order to enrich the generated data with context information. It is mainly composed of four services (as presented in Figure 4.2):

(i) the *semantic enrichment* that adds meaning to the data based on the ontology;

(ii) the *data normalization* that unifies the data values related to the same observed parameters;

(iii) the *data cleaning* that plays an important role in improving the quality of data by removing corrupt or inaccurate records such as those resulting from the misbehavior of sensors;

(iv) and the *data storage* that stores the prepared data in big data clusters to be exploited by the analytics and the visualization services.

The analytic services operate on the harmonized datasets to discover new knowledge related to the domain based on scalable machine learning algorithms, MapReduce jobs and other big data technologies for distributed large datasets processing. The results of the analytic services will be annotated using the "*Semantic Value Annotation*" service in order to expose the generated knowledge as a service to the appropriate consumers or to be reused by the management processes. Furthermore, our KaaS offers also visualization services to graphically represent data and the analytics results in order to facilitate the interpretation for domain experts. The access to our KaaS services is based on access control policies.

In the next section, we specialize the KaaS in healthcare and we represent an ontological model within the *Semantic Knowledge Layer*, which represents the foremost element for the integration of heterogeneous data stemming from multiple wearable devices for the patient health management.

4.4.2 Wearable Healthcare Ontology

Our proposed *Wearable Healthcare Ontology* (*WH_O*)[160] is designed to deal with the heterogeneity of wearable data to ensure semantic interoperability and

allow the system perceiving and understanding the received data in order to generate more accurate knowledge about the patient such as detecting and/or predicting anomalies. The WH_O characterizes the wearable devices and their generated data to provide a smart system for patient care management, independent of the disease. Figure 4.3 depicts the main classes of the WH_O providing a common representation and sharing the same meaning among the different wearable devices. An ontological representation of WH_O using protégé is presented in Figure A.2.

Based on our state of the art, we found that the IoT-O [37] is the most generic representation of IoT, based on wide range of upper ontologies, and supports the autonomic management through describing the sensors and actuators. However, within IoT-O, the sensors' observations are stored in the ontology, while in our work we propose to store the observed data in NoSQL databases. Consequently, we define data processing services that harmonize the data based on wearables description. To this end, we propose to reuse and extend IoT-O to support the wearable healthcare management. For clarity reasons, in Figure 4.3, we present only the main concepts of IoT-O that have been aligned with WH_O. A wearable device is considered as a sub-class of the Sensor class in IoT-O. In this way, we reuse: (i) the Sensor Model of IoT-O to describe the wearable capabilities, (ii) the Service Model to describe the wearable services and methods, and (iii) the Actuator Model for the autonomic management. We replace the Observation Model with the *Wearable Healthcare Model* which describes the sensory model (blue color) and the patient context (yellow color).

In WH_O, each patient may be equipped with a set of wearable devices having specific configuration that depends on the patient medical conditions and disease severity. Consequently, each patient has at least one "WearableConfiguration" which is specific to a wearable device. Each "WearableConfiguration" is characterized by its "StartDate" and "EndDate", as well as the frequency of measurement (e.g. 1 measurement per day). Each wearable may implement at least one "MeasurementService" which is responsible for measuring a parameter such as "Blood Sugar", "Blood Pressure", "Heart Rate", "Step", etc. Each "MeasurementService" expresses the generated value with a specific unit, and stores the observed data in datasets identified through their endpoints. Each unit may correspond to another equivalent unit measuring the same parameter. The measured parameters allow identifying medical conditions (disease, symptom, etc.). For example, the "Hypertension" is associated to monitoring the "Blood Pressure". Semantically characterizing this piece of information fosters the dynamic discovery and deduction of new medical conditions that may affect the patient.

To provide a personalized management of the patient health, each patient should have his/her own "TargetGoal" which depends on the patient medical characteristics and disease stage. This goal, which is fixed by the physician, is the clue for detecting anomalies.

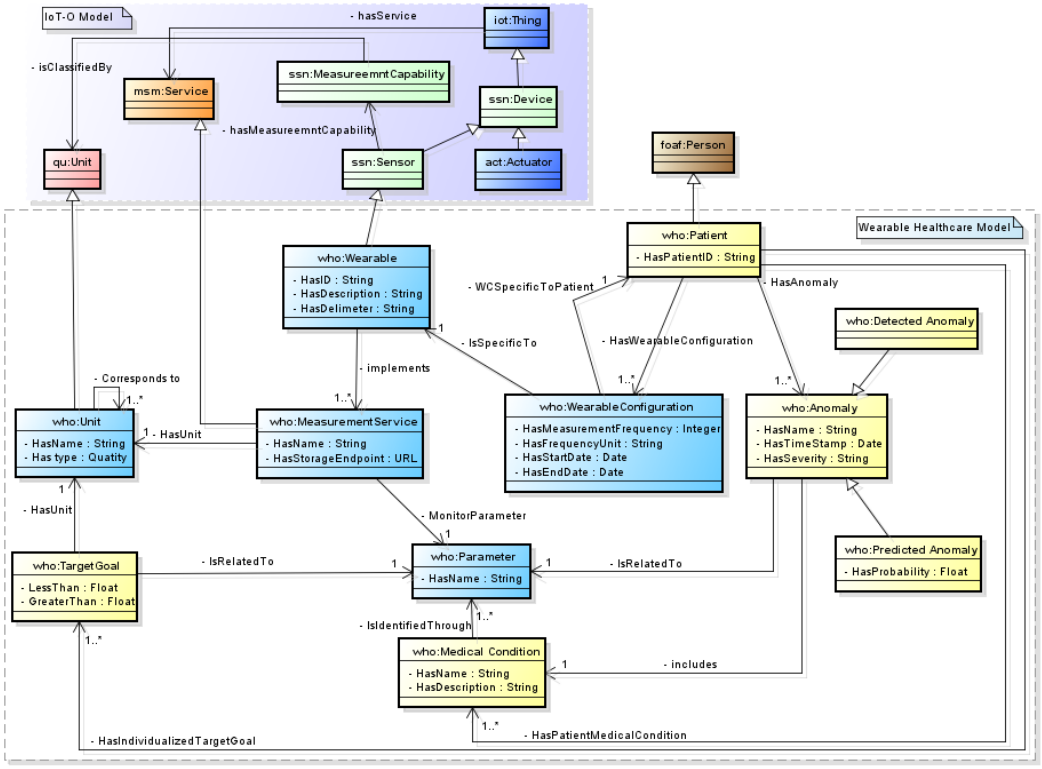

Figure 4.3: Depiction of the main classes and relationships of the Wearable Healthcare Ontology

In this way, the system supports the specification of personalized anomalies depending on each patient' characteristics and data generated by the wearables without modifying the analytic services. The WH_O identifies two main types of anomalies: "Detected Anomaly" such as detecting an "increased glucose level" based on rules, and "Predicted Anomaly" such as predicting "hypertension" based on machine learning algorithms operating on datasets of patients having diabetes (with and without hypertension) [141]. If an anomaly is detected or predicted, it will be automatically populated in the KaaS. According to its severity, notifications will be sent to the appropriate physicians to keep them up-to-date about the patient health evolution and avoid complications by taking preventive and timely interventions.

It is worth noting that the interactions among the different components of the KaaS are driven by the patterns that have been proposed in chapter 3 for the development of Autonomic and Cognitive IoT-based systems. Thus, we propose in the next section an instantiation of these patterns to design and develop the KaaS platform specific model for the near real-time management of the patient health evolution.

4.4.3 A Cognitive Monitoring System for Patient Health Management

We propose a *Cognitive Monitoring System* to timely manage the patient health evolution, based on data stemming from multiple wearable devices. To this end, we instantiated and integrated mainly three patterns presented in chapter 3. The implementation of the proposed patterns will be offered as services through following the proposed KaaS architecture. The adopted patterns are:

(i) the *Cognitive Monitoring Management* pattern in order to identify the interactions among the wearable devices and the physicians;

(ii) the *Semantic Knowledge Mediator* pattern in order to manage data heterogeneity and enable the collaboration and interaction among the wearable devices and the physicians;

(iii) and the *Big Data Stream Detection* pattern, which is an evolution of the *Cognitive Monitoring Management* pattern supporting the big data stream processing, in order to provide near real-time visualization and personalized detection while managing data heterogeneity and velocity.

Based on a user-friendly interface, we developed a collaborative semantic web platform, implementing the *Semantic Knowledge Mediator* pattern, to allow acquiring the characteristics of the used wearable devices based on the WH_O. It is built on the top of Semantic MediaWiki[7](SMW), which is a collaborative semantic authoring tool. SMW unleashes the power of wikis for collaborative knowledge management, and ontologies for providing a common understanding of the domain. It offers mapping mechanisms for formalizing annotations embedded in wiki pages into OWL DL ontology language [98]. Originally, SMW stores the annotations in SQL database (e.g. MySQL). Thus, to leverage SPARQL querying language, we extend our platform with Apache Fuseki[8] in order to store annotations in RDF format. Thus, our platform provides a SPARQL endpoint that allows the management processes automatically seeking the appropriate information. Figure 4.4 represents a snapshot of the user interface when annotating the wearable devices' characteristics. Likewise, the platform is used by the physicians to annotate the patient' context and specify his/her target goal for each monitored parameter. As previously mentioned, the generated annotations represent mainly the sensory and context knowledge that will be used in the *Big Data Stream Detection* pattern when processing the data.

In the *Big Data Stream Detection* pattern, the monitoring refers to collecting the data, normalizing and storing it in distributed clusters, while the analysis process refers to detecting personalized anomalies. Figure 4.5 illustrates the workflow

[7]https://www.semantic-mediawiki.org/wiki/Semantic_MediaWiki
[8]https://jena.apache.org/documentation/serving_data/

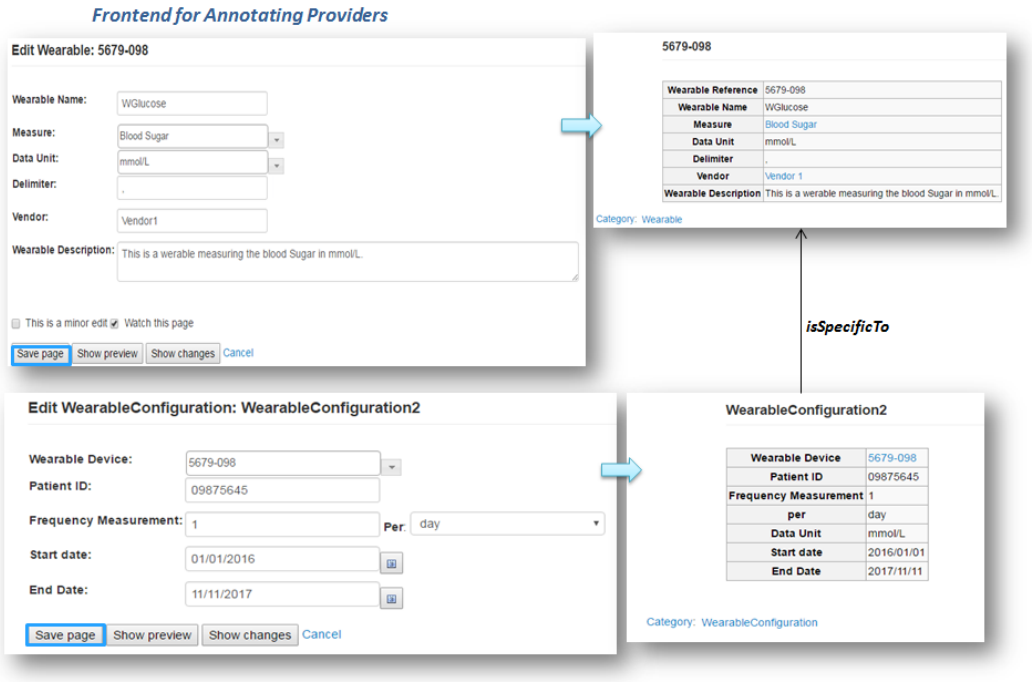

Figure 4.4: An expert of the collaborative semantic web platform for providers' characteristics annotation

and the interactions among the different components for the development of the big data stream detection within the KaaS. Once the wearable devices are subscribed to the semantic platform through annotating their characteristics based on WH_O, they will be configured to send asynchronous data to Apache Kafka[9], which is a scalable and high-throughput distributed publish/subscribe messaging system. The wearable devices are considered as data producers (publishers). And, we propose to use Apache Storm[10] to consume the data from the Kafka topics based on Kafka-storm connector. We defined a storm topology that encodes the data workflow processing using three bolts for (1) data normalization, (2') data storage in distributed system to be analyzed, and (2") near real-time problem detection. Both data storage and problem detection are executed in parallel. In the context of managing diabetes, we assume that we have wearable devices that measure the "Blood Sugar", but group of them express the measured data in mg/dL while the rest in mmol/L. Herein, an example of the received stream data from W1ID (mg/dL) and W2ID (mmol/L):

W1ID,timestamp,170
W2ID; timestamp;11

[9]http://kafka.apache.org/
[10]http://storm.apache.org/

Figure 4.5: Big data stream detection sequence diagram within the KaaS

The **Normalization** bolt refers to the *Sensor Cache* where information about the active wearable unit and delimiter are retrieved from Fuseki through enacting the query presented in Listing 4.1. After that, it refers again to the *Sensor Cache* to extract the unit, and decide if the measured value should be converted or not. Once data is normalized, it will be sent in parallel to the Hadoop's HDFS system[11], and to the **Detection** bolt. The harmonized data stored in HDFS can be visualized, and also used by advanced analytic services implemented using Apache Spark and MapReduce framework.

Listing 4.1: Retrieve the units and delimiter of active wearables

```
PREFIX prop:<http://192.168.0.104/Healthcare/index.php/Special:
   URIResolver/Property-3A>
Select distinct ?wid ?unit ?delim where
{       ?wc prop:IsSpecificTo ?w.
        ?w prop:HasID ?wid.
        ?w prop:Implements ?ms.
        ?ms prop:HasUnit ?unit.
        ?ms prop:HasDelimiter ?delim.
        ?ms prop:MonitorParameter ?param.
        ?param prop:HasName ?paramName.
        ?wc prop:HasStartDate ?sd.
        ?wc prop:HasEndDate ?ed.
        Filter (?paramName ="topic").#such as Blood Sugar
        Filter (?sd<= "date"). # date when updating the cache
        Filter (?ed>="date").
}
```

The **Detection** bolt refers to the patient *Goal Cache* in order to detect personalized anomalies according to the fixed patient target goal. The *Goal Cache* is dynamically up-to-date through enacting the SPARQL query presented in Listing 4.2. If a problem is detected, it will be populated in the SMW platform, and a notification is sent to the appropriate physician.

Listing 4.2: Retrieve the list of patients with their target goal

```
prop:<http://192.168.0.104/Healthcare/index.php/Special:
   URIResolver/Property-3A>
Select ?ID ?m where
{       ?p prop:HasIndividualizedTargetGoal ?goal.
        ?goal prop:IsRelatedTo ?param.
        ?goal prop:LessThan ?m.
        ?param prop:HasName ?lab.
        Filter (?lab="ParamName").# for example, Blood Sugar
        ?p prop:HasPatientID ?ID.
}
```

[11] http://hadoop.apache.org/

In the next section, we evaluate the performance of the proposed system within the KaaS for the near real-time detection following different deployment configurations in order to appraise the system response time and scalability as well as its cost.

4.5 Performance Evaluation

To evaluate the proposed cognitive monitoring system performance within the KaaS, let's consider different patients with type 2 diabetes equipped with heterogeneous wearable devices measuring the blood sugar parameter. These wearable devices are connected and characterized in the KaaS. In turns, the patient medical conditions and target goals are also updated in the KaaS. To simulate the wearable devices' behavior, we inspired from the data published in [161]. Thus, we extract an expert of this data expressed in mg/dL, modify the data structure and add some fields such the wearable ID, the delimiter, etc. We converted some of them to mmol/L to create heterogeneous data. Then, we duplicated the data to create different scenarios. These scenarios have been tested in different KaaS configurations including three infrastructures with two different modes related to the Storm topology configuration. We used the ArchiMate standard to represent the business, application and technology layers of each KaaS configuration.

To conclude, the evaluation process has been conducted as follows:

- At a first stage, we evaluate the system performance following the on-premises deployment, and compare it to the results obtained when applying the multi-tenant management process pattern on the cloud.
- At a second stage, we propose a multi-node deployment of the system components in the cloud –fully distributed system. We evaluate the response time when parallelizing the data processing over 2 workers in two different cloud infrastructure configurations. Moreover, we deduce for each cloud configuration the cost function in terms of CPU consumption.

Our goal is not to find the optimal configuration but to illustrate that thanks to our flexible architecture, we can dynamically allocate the required resources and instantiate the needed distributed components in order to cope with a large diversity of non-functional requirements.

4.5.1 On-premises vs Cloud Evaluation

In this evaluation, we use two different infrastructures: a local machine for the on-premises deployment and a cloud server for the cloud deployment.

First, we ran the experiments on the following machine configuration: memory 12GB; processor Intel(R) Core(TM) i7-2640M CPU @ 2.80GHz; and 64-bit Windows 7 operating system. We installed Proxmox[12] on this machine for virtualization. Proxmox allows easily exporting the virtual containers to be hosted in a cloud environment and to benefit from the elasticity of the cloud. We created two virtual containers hosting Ubuntu operating system, which play the role of distributed machines. We attributed 6 GB of memory to Proxmox distributed as follows: (Fuseki, 4GB) and (Semantic MediaWiki, 2GB). Moreover, we installed another virtual machine including Ubuntu operating system and consuming 8GB of memory. Within this machine, we installed Apache Kafka, Apache Hadoop, Apache Storm and Apache Zookeeper (to coordinate Kafka and Storm instances). Due to memory constraints, only the Fuseki container is running. The deployment of the proposed architecture is portrayed in Figure 4.6.

We varied the number of incoming streams in order to evaluate both the system response time and scalability. It is worth mentioning that all the data stream providers (connected devices) are using the same instance of the consumer (storm topology executed through 1 worker).

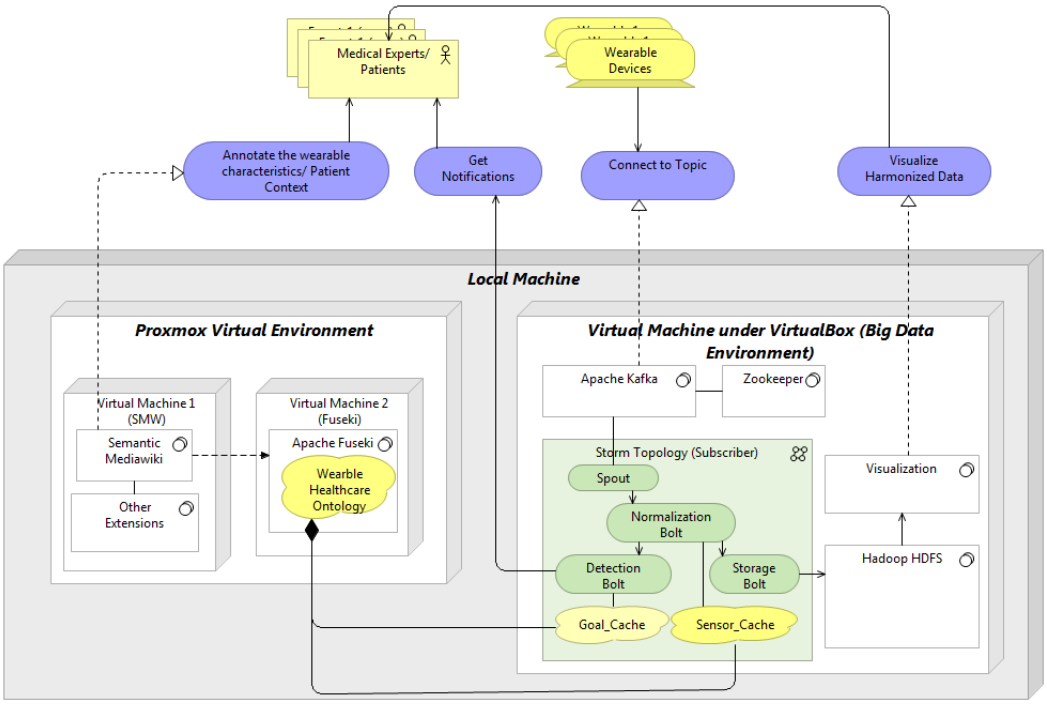

Figure 4.6: On-premises deployment of the cognitive monitoring system

Figure 4.7 portrays the response time in seconds of each executed scenario. We

[12]Proxmox: https://www.proxmox.com/en/

noted that a delay is reported when big number of requests is sent to the platform. It took around 4545 seconds ≈ 75 minutes to process 600,000 data streams, which is not adequate for healthcare applications where timely response is required to manage the patient health. For instance, if the system is managing patients with severe diabetes, it is required to provide near real-time processing to intervene at the right time when the glucose drop too low. Moreover, we noted that for 1,200,000 data streams, a bottleneck is reported in the kafka broker. These results strongly depend on the hardware configuration on which the experiment is conducted as well as the deployment of the system components.

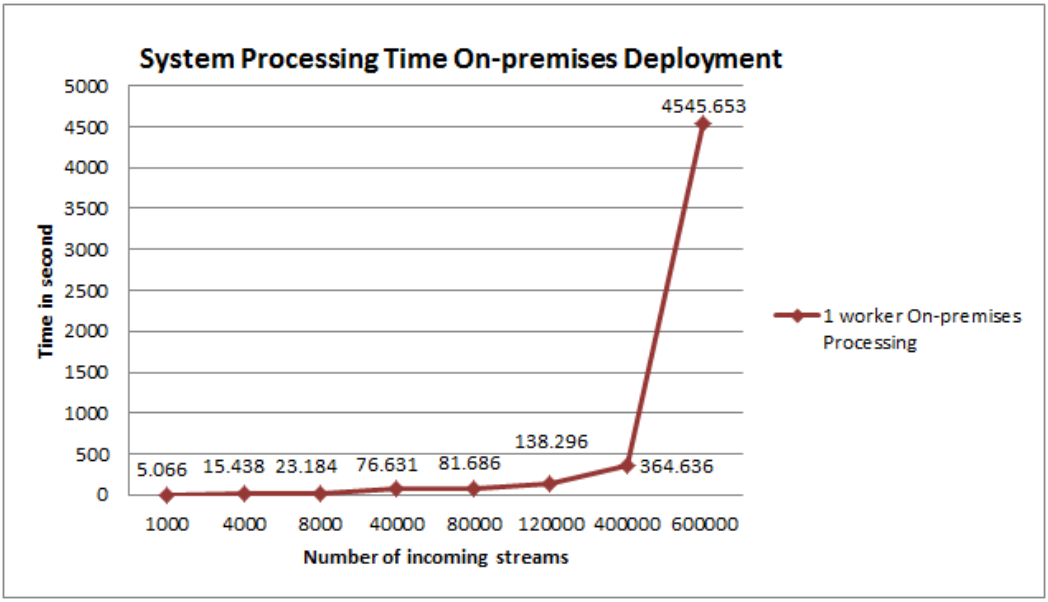

Figure 4.7: Performance evaluation on-premises deployment

Consequently, to provide better system performance as well as near real-time data processing, we propose to apply the *Multi-tenant Management Process* pattern in order to benefit from the scalability, multi-tenancy and IT resources virtualization on the cloud. We refer to the following cloud server infrastructure (cloud-config1): 8 Intel(R) Xeon(R) CPU D-1521@ 2.40GHz, 32 GB of memory, and 1.77 TiB of disk storage.

We used proxmox as a virtualization environment, and we created four linux containers (LXC) (106, 110, 107 and 108). The "container 106" contains SMW associated to a set of extensions in order to offer a collaborative user-friendly interface for semantic annotations, while the "container 110" contains the Fuseki to store the semantic annotations based on the *Wearable Healthcare Ontology*. Both of these two containers constitute the implementation of the *Semantic Knowledge*

Mediator pattern. The "container 107" deploys Storm, Zookeeper and Kafka for data collection, harmonization and anomaly detection, while the "container 108" deploys Hadoop for storing the harmonized data, visualizing as well as analyzing the data based on MapReduce jobs and other big data frameworks. Figure 4.8 describes the deployment process following the *Multi-tenant Management Process* pattern.

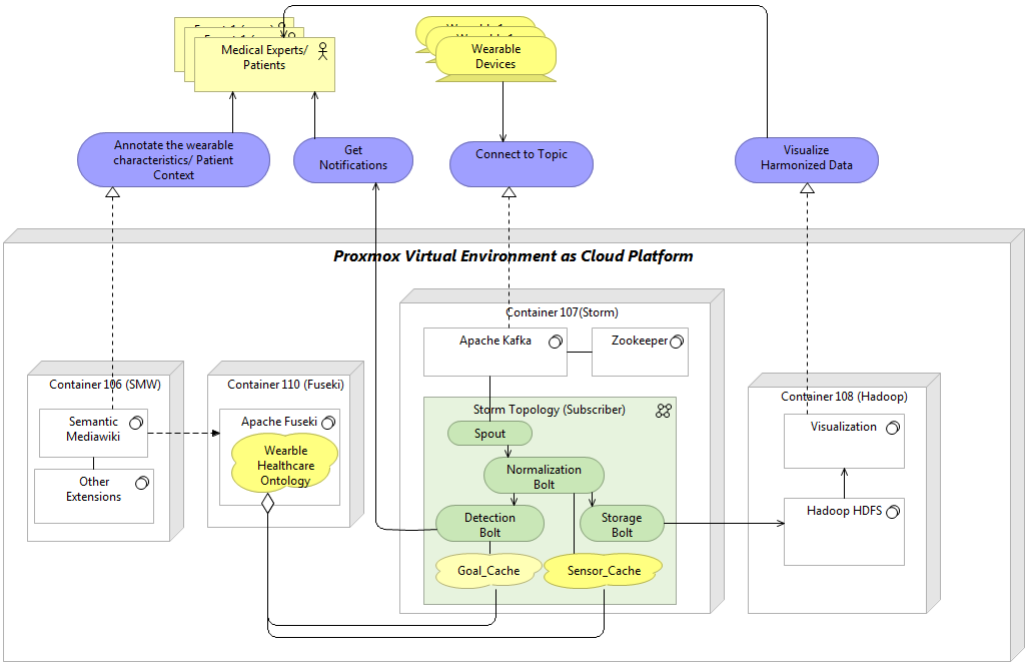

Figure 4.8: Cognitive monitoring system deployment in cloud-config1

To evaluate the gain of this architecture, we run the same testing scenarios, which have been already conducted in the on-premises deployment, on the cloud-config1. We run the storm topology implementing 1 worker on a Storm local cluster, then, we measured the processing time and the ability of the platform to process all the incoming streams. Table 4.2 portrays the measurements. We noted a huge gap between the cloud deployment and on-premises deployment in terms of processing time and scalability management.

Figure 4.9 delineates the gain that results from offering the data processing and detection as a service. Unlike the on-premises deployment, on the cloud, the system is able to successfully process all the 1,200,000 received streams in around 6 minutes without noting any bottleneck or stream lost, which is crucial in patient management systems. Indeed, each data stream should be successfully processed since it may include important information about the managed patient that may impact on the decision process. For instance, let's consider a big number of patients

managed by various wearable devices measuring the blood sugar and the blood pressure. If the system fails to process one of the received streams that may indicate that the diabetic patient is developing hypertension, this may lead to health complication when adapting the diabetes treatment, because some diabetic drugs may amplify the hypertension condition.

	Processing Time in second	
Number of incoming streams	**1 worker On-premises**	**1 worker on cloud**
1000	5.066	0.65
4000	15.438	1.958
8000	23.184	4.769
40000	76.631	16.646
80000	81.686	26.899
120000	138.296	42.294
400000	364.636	127.142
600000	4545.653	181.972
1200000	Bottleneck	384.233

Table 4.2: Performance measurements on-premises vs cloud deployment

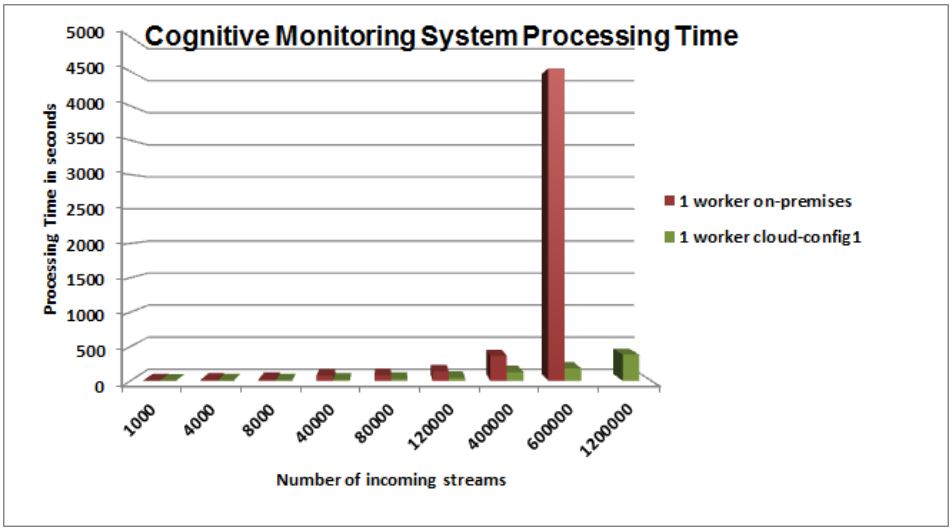

Figure 4.9: Comparison of the system performance on-premises vs cloud deployment

It is axiomatic that the cloud deployment (application of the Multi-tenant Management Process pattern) potentially contributes to providing better scalability and response time compared to the on-premises deployment. However, we noted that a drawback is adrift from this deployment architecture. It concerns the ability of the system to keep processing data if the "container 107" stops working or crashes. Thus, centralizing the deployment of Kafka and Storm with a local cluster deployment mode of Storm is not recommended. Consequently, we propose a multi-node deployment of the cognitive monitoring system within the KaaS in order to enact the storm topology on a production cluster (Master/Slave deployment).

4.5.2 Multi-Node Cloud Evaluation

To fully leverage the distribution and the parallelism on the cloud, Figure 4.10 proposes a new deployment of the proposed cognitive monitoring system for data stream processing. Unlike the architecture described in Figure 4.8, Storm is deployed over three containers following the master/slave pattern. The "container 113" contains the master storm running the nimbus daemon, while two storm slaves are deployed in the "container 114" and the "container 115" to run the topology workers and threads. The coordination of the master and the supervisors is controlled by Zookeeper. Both Kafka and Zookeeper are installed in the "container 111". Figure 4.10 describes the allocated IT resources for each container based on cloud-config1. Data providers are deployed in other containers and are sending asynchronous messages to the Kafka broker.

Following this architecture, we run the storm topology in production cluster with two different configurations:

- *1 worker*: each bolt is executed by only one thread. As two storm supervisors are deployed, only one supervisor is selected to run the topology.
- *2 workers*: This configuration supports the parallelism when processing the data streams in a distributed way over the two deployed supervisors. To increase the parallelism, we set the **parallelism_hint** of the Normalization bolt and Storage bolt to 2.

We evaluated both configurations' performance through running the aforementioned testing scenarios. The measurement of the processing time for each configuration is illustrated in Table 4.3, while a visual representation of the performance is represented in Figure 4.11. We noted that the parallelism contributes to reducing the processing time, especially when processing a big number of data streams. For instance when processing 1,200,000 data streams in cloud Config1, with the parallelism, we gained 53 seconds which is an important value for near real-time applications.

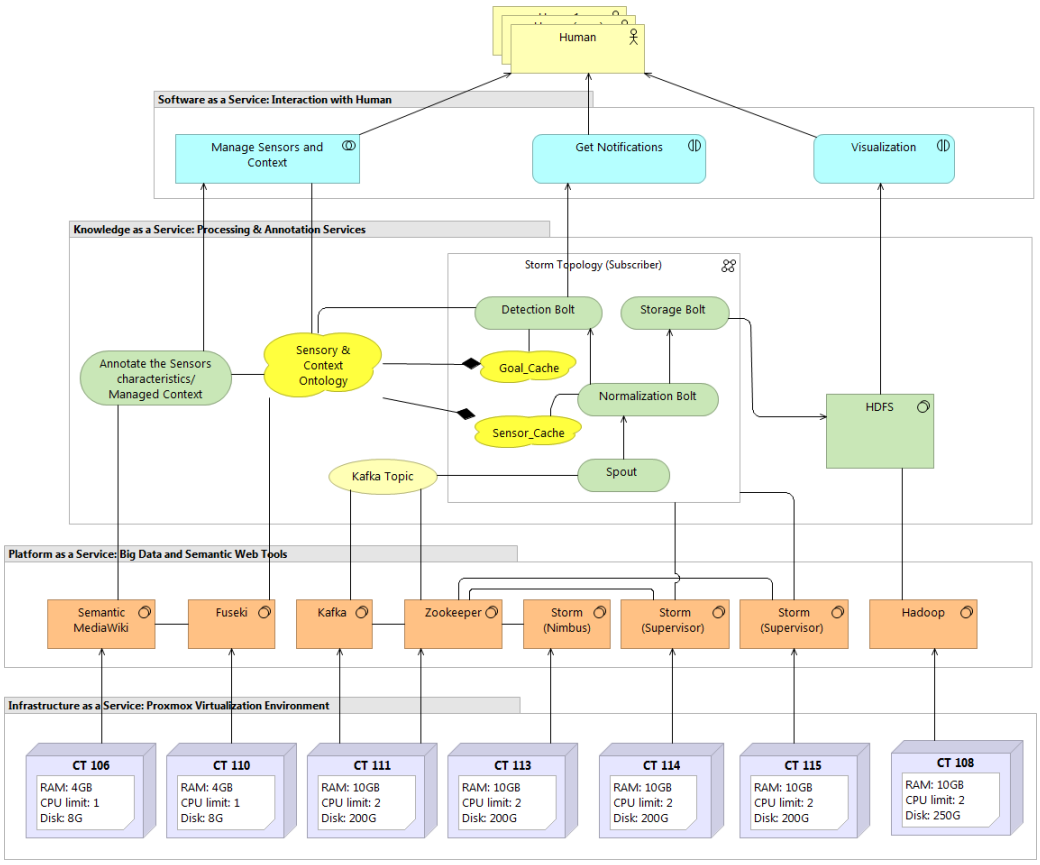

Figure 4.10: A multi-node deployment on the cloud

	Processing Time in second (cloud-config1)	
Number of incoming streams	**1 worker**	**2 workers**
1000	1.497	1.251
4000	4.174	4.081
8000	6.485	6.180
40000	17.281	15.668
80000	28.682	27.410
120000	43.740	39.775
400000	132.685	118.348
600000	194.636	170.841
1200000	390.200	337.745

Table 4.3: Performance evaluation of the parallelism on the cloud

Despite the processing time, we measured the function cost in terms of CPU consumption from the Storm's supervisors side (CT114 and CT115), since the topology is running on these containers. Figure 4.12 provides an overview of the CPU consumption when running only *1 worker* on CT115, and when running *2 workers* on both containers CT114 and CT115. We noted that the execution of 2 workers over 2 containers provide faster processing time than using only 1 worker. However, when running 2 workers on storm production mode, each container may consume an average of 30% of the allocated resources (2CPU for each one), while when running 1 worker, only one container is running and its CPU consumption may reach 35% of the allocated resource (2CPU). Thus, the parallelism provides better performance, but requires more CPU resources.

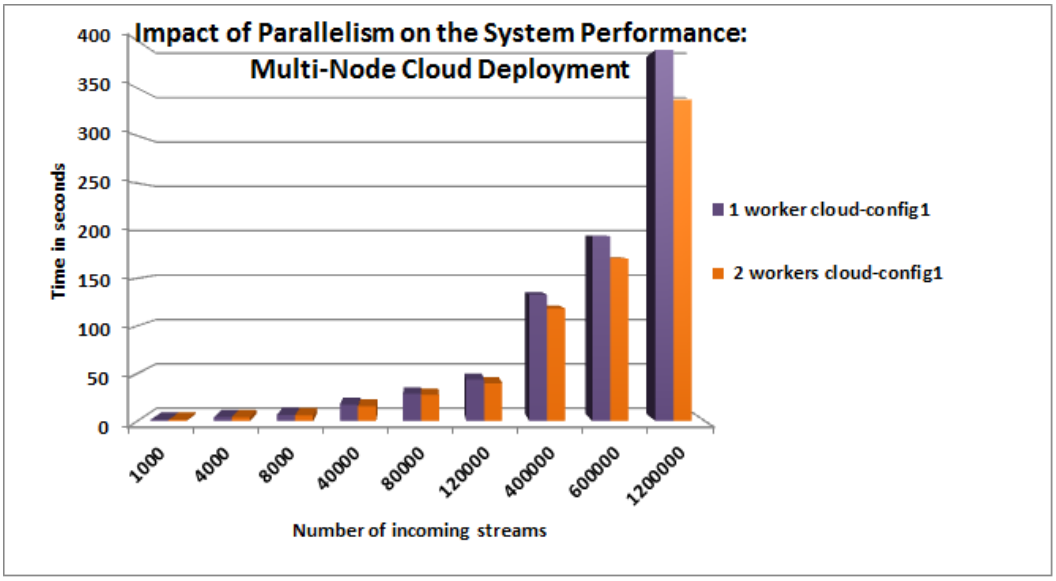

Figure 4.11: Performance evaluation of the cognitive monitoring system in production mode

At a second stage, we increased the allocated CPU units of CT114 and CT115 to 4 units and the CT111 to 3 units in a new cloud server that has the following configuration (cloud-config2): 8 Intel(R) Xeon(R) CPU D-1521@ 2.40GHz, 128 GB of memory, and 1.77 TiB of disk storage. After deploying the different containers on cloud-config2, we evaluated again the same testing scenarios, and we compared the performance of the system when implementing *2 workers* to what we got in cloud-config1. Table 4.4 and Figure 4.13 portray the obtained results. We noted that increasing the CPU contributes to reducing the processing time, especially when processing big number of incoming data streams. For instance, when

Figure 4.12: CPU consumption in the Storm supervisors when processing 1,200,000 streams

receiving 1,200,000 data streams, we gained up to 30 seconds which is considered as gain for near real-time systems.

	Processing Time in second	
Number of incoming streams	**2Workers cloud-config1**	**2Workers cloud-config2**
1000	1.251	1.285
4000	4.081	4.080
8000	6.180	5.247
40000	15.668	14.623
80000	27.410	27.247
120000	39.775	34.861
400000	118.348	111.443
600000	170.841	163.003
1200000	337.745	301.639

Table 4.4: The cognitive monitoring system performance measurements on two cloud infrastructures

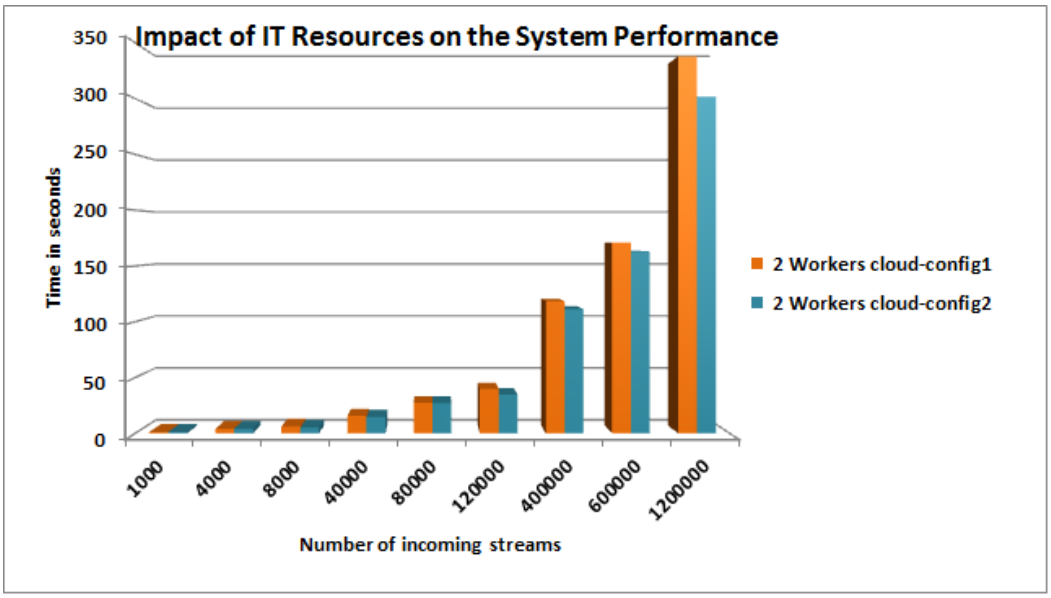

Figure 4.13: Evaluation of the impact of IT resources configuration on system performance

To conclude, choosing the appropriate cloud infrastructure and the system deployment configuration within the KaaS depend on the application requirements and the availability of IT resources. For instance, if the system is used to manage patients with chronic disease at early stage to prevent the patient health degradation, the 30 seconds that we gained in cloud-config2 is not considered as a delay in cloud-config1. In this case, the function cost will be used to select the right configuration. However, if the system is conceived for managing patients with severe clinical conditions, near real-time detection is mandatory. In this case, the distributed architecture deployed on cloud-config2 is more adequate.

4.6 Conclusion

In this chapter, we provided a *Knowledge as a Service* platform that extends the NIST cloud computing and NIST big data reference architectures with a semantic layer in order to cover the heterogeneity of data stemming from wearable devices, while guaranteeing the system scalability. The proposed KaaS platform will be used to deploy the management processes, knowledge components and databases based on the proposed patterns in chapter 3.

As an application, we referred to managing patient health evolution based on wearable devices. Thus, we proposed a cognitive monitoring system implementing the *Cognitive Monitoring* pattern, the *Semantic Knowledge Mediator* pattern, the *Big Data Stream Detection* pattern and the *Multi-tenant Management Process* pattern in order to provide a scalable interactive IoT-based system for patient health management. To this end, we elaborated the WH_O that semantically describes the wearable devices for the integration of healthcare data. Based on WH_O, the system is able to process heterogeneous data stream independently of their structure. An implementation of the proposed system is provided based on big data and semantic web technologies and we evaluated its performance as well as the associated cost based on different cloud infrastructures and configurations. The main goal of the proposed evaluation is illustrating that the non-functional requirements of diverse healthcare scenarios can be satisfied by the the dynamic configuration of the KaaS where the monitoring cognitive system is deployed.

In the next chapter, we propose to enhance our *Cognitive Monitoring system* with new components including management process and cognitive components in order to evolve to a *Prescriptive Cognitive system* able to assist the physicians in solving problems when personalizing the patient treatment.

CHAPTER 5

A Prescriptive Cognitive System for Patient Treatment Management

"A good decision is based on knowledge and not on numbers."

- Plato

Contents

5.1 Introduction 110

5.2 Decision-making for Treatment Management 110

5.2.1 Clinical Decision Support System 111

5.2.2 Treatment Adaptation & Personalization 111

5.2.3 Linked Data in Healthcare 114

5.2.4 Discussion 115

5.3 A Prescriptive Cognitive System for Personalization 116

5.3.1 System Architecture Overview 116

5.3.2 A Collaborative Methodology for Medical Knowledge Formalization 117

5.3.3 Treatment Plan Ontology 120

5.3.4 A Collaborative Semantic Web Platform for Medical Knowledge Acquisition 123

5.3.5 Ontology-based Planning Algorithm for Treatment Personalization 126

5.4 Clinical Evaluation 129

5.4.1 Use Case 1 129

5.4.2 Use Case 2 130

5.4.3 Use Case 3 from Clinical Diabetes Journal 131

5.5 Performance Evaluation 133

5.6 Conclusion 137

5.1 Introduction

In the previous chapter, we highlighted the benefits of using wearable technologies for the management of patient health evolution to provide personalized anomalies detection at the right time. Nevertheless, IoT-based systems' capabilities can go further to support reasoning and smart management of patient treatment. In healthcare, the personalization of patient treatment requires the integration of different types of knowledge such as the used strategies to manage a particular disease (procedural knowledge), the drug characteristics, side effects, etc. Advances in web technologies foster sharing medical knowledge. But, integrating these knowledge sources is not straightforward and costly for physicians, due to their distribution, heterogeneity and huge volume. Acquiring the IoT systems the ability to reason on these knowledge sources requires understanding and interpreting its meaning.

Consequently, in this chapter, we extend the system proposed in the previous chapter with new capabilities in order to reach the *prescriptive cognitive maturity level* where the IoT-based system is able to provide recommendations to physicians. Thus, first, we discuss existing works dealing with treatment adaptation and personalization. Through collaborating with medical experts, we propose a methodology for extracting and formalizing the treatment plan based on ontologies. The output is a flexible generic semantic model named *Treatment Plan Ontology (TPO)* describing the medical interventions and their granular characteristics that underpin the adaptation of the patient treatment. To facilitate the interaction with medical experts, we develop a collaborative semantic web platform that allows the medical experts transferring their knowledge based on TPO, and exposing it as linked data to automate decision making. The next step focuses on integrating TPO with external reusable medical knowledge sources in order to provide personalized treatment management. Hence, we propose an ontology-based planning algorithm that aggregates TPO with DrugBank to detect drug-drug interaction, especially in case of comorbidity where different diseases are presented. Finally, we evaluate the proposed plan process from the clinical perspective through managing hyperglycemia in type 2 diabetes, and we highlight its performance on the cloud.

5.2 Decision-making for Treatment Management

In this section, we introduce the clinical decision support system. Then, we identify medical knowledge sources that can be reused in the personalization process. After that, we survey recent works dealing with the treatment adaptation and highlight the main used approaches and techniques.

5.2.1 Clinical Decision Support System

Clinical decision support systems (CDSS) have shown great efficiency in managing patients' health and providing personalized treatment based on information technologies. CDSS may focus on sharing the patient profile including the treatment history, the observations, etc., diagnosing patient health, and generating clinical recommendations based on evidence-based medicine and clinical guidelines [162]. The development of CDSS is classified into two categories [163]:

- Non-knowledge-based CDSS which uses machine learning and other statistical pattern recognition approaches that allow the computer automatically learning from past experiences and/or detecting patterns from the clinical data;
- Knowledge-based CDSS relies on the conceptualization of the medical knowledge encoded in a computer interpretable format, called computer-interpretable guidelines (CIGs) [164] as well as the reasoning process in order to elucidates and facilitates the decision-making.

However, current CIG languages are not flexible enough to support the integration of external knowledge sources, the dynamic knowledge update, and the patient treatment personalization based on context changes [165]. Few studies focus on the dynamic adaptation of the patient treatment based on the context changes. Indeed, treatment adaptation requires more flexible representation of the medical knowledge on which the management processes operate. Generating personalized treatment should take into consideration both the changes of the patient conditions and environment factors that influence response to therapy, as well as the medical knowledge including reliable accessible sources. In the next section, we focus on the treatment adaptation: the approaches used to model the medical knowledge, and the methods used to select recommendations.

5.2.2 Treatment Adaptation & Personalization

Various approaches have been used to represent the medical knowledge and adapt the patient treatment. Table 5.1 classifies the harvested works and delineates the used techniques for recommendation as well as the methods used to formalize the medical knowledge.

Huang et al. [166] proposed a recommendation service that implements a sophisticated algorithm based on mathematical model and data mining technique to match patient status with medical interventions and calculates the recommendations ratings. Ferrer et al. [167] mapped the clinical guideline to the Asbru model and then to the HPDL in order to automatically generate the patient care pathway based on the Hierarchical Task Network (HTN) planning technique. However, the

proposed approach does not take into consideration the dynamic adaptation of the care pathway based on the context changes. Similar to Ferrer et al. [167], Millan et al. [168] used the HTN for generating recommendations. But, Millan et al. [168] contributed in the dynamic adaptation through using the PELEA architecture, which is a continuous planning approach. Thus, based on the monitoring and the detection of deviations, their system is able to adapt the care pathway through extracting information related to patient and resources from the Virtual Medical Record (VMR). As a continuity of this work, Sánchez-Garzón et al. [169] integrated the monitoring of patients' daily activities in an ambient environment. Thus, based on the collected data about the patient and the Clinical Practice Guidelines, the system is able to repair the current plan and to automatically update the medical record. However, Huang et al. [166], Ferrer et al. [167] and Millan et al. [168] did not consider the medical knowledge evolution and the dynamic update of the decision rules since the planning is based on use cases, or on problems and solutions described in text files.

In this context, ontologies have been widely used for formalizing the medical knowledge and reasoning to generate recommendations. We cite for instance the work of Riaño et al. [170] which provided a complex personalized care ontology for chronic disease namely Case Profile Ontology (CPO) in the context of the K4CARE project[1]. The adaptation is realized by instantiating the specific CPO ontology according to the patient profile, then, extracting individualized decisions from the general formal interventions designed with SDA* model. Emerencia et al. [171] proposed wegweis, which is a web-based advice platform, for managing patients with schizophrenia. Wegweis provides behavioral advices based on a problem severity and advice priority through a selection and ranking algorithms. The authors used the ontology to decouple the problem from the advice and to add robustness by inferring advices of hierarchical problems. Likewise, Grando et al. [165] proposed a state-based goal framework implemented within the argumentation technique to represent medical decision within guideline. The association among goals, tasks, argument, belief, temporal constraints, actors and patient are designed with ontology, and the selection of the decision is achieved through rules and an aggregation function based on the argument-weight. The ontology classes and rules are implemented in the COGENT system which provides a graphical interface to define models and context, and to simulate the execution of the model.

Riaño et al. [170], Emerencia et al. [171] and Grando et al. [165] proposed expert systems that do not support the dynamic adaptation, they are limited to the medical knowledge formalization and proposing decision rules.

[1] K4CARE: http://www.k4care.net/

Related Works	Dynamic Adaptation	Medical Knowledge Sources	Medical Knowledge Representation	Recommendation Technique
Huang et al. [166]	No	Clinical Cases	-	Data Mining
Ferrer et al. [167]	No	Clinical Guideline	Text File (HPDL)	HTN Planning
Millan et al. [168]	Yes	Clinical Guideline	Text File (HPDL)	HTN Planning
Riaño et al. [170]	No	Clinical Guideline Experts	SDA* model	-
Emerencia et al. [171]	No	Experts	OWL (Protégé)	Selection and ranking algorithm
Grando et al. [165]	No	Clinical Guideline	GOGENT language (classes and rules)	Rule based
Alexandrou et al. [172]	Yes	Experts	OWL (Protégé)	SWRL rules
Yao and Kumar [173]	No	-	OWL (Protégé)	SWRL rules
Lasierra et al. [36]	Only detection	the X73 standard (sensors) + Physicians feedbacks	OWL-DL (Protégé)	-

Table 5.1: A survey on treatment adaptation and personalization

In this context, Alexandrou et al. [172] proposed an ontology based solution able to automatically observe the execution of each task specific to a patient treatment workflow process and to dynamically adapt the treatment schema concerning the healthcare business process. The proposed ontology is named Adaptive Clinical Pathway Ontology. It is implemented in OWL and the medical knowledge is acquired by experts using Protégé, while the adaptation process is based on the Semantic Web Rule Language (SWRL). Similar to Alexandrou et al. [172], Yao and Kumar [173] proposed to use Protégé to acquire the clinical context ontology and SWRL rules to generate recommendations based on the patient context changes. Yao and Kumar [173] identified a wider work which supports the patient treatment generation and prescription checking based on rules. However, these rules are static and hard-coded. Thus, if new medical knowledge is updated, new rules should be encoded, and/or existing ones should be updated. Lasierra et al. [36] presented a relevant work that relies on the autonomic computing to manage patients with chronic disease. The authors used ontology to enable semantic interoperability when monitoring the patient at home based on medical sensors. However, their work is limited to the monitoring and analysis processes in order to identify alarms and send warnings to physicians and patients.

5.2.3 Linked Data in Healthcare

Linked data refers to enabling the extension of the Web with a global data space based on open standards –the Web of Data. It provides a common data model that makes it possible to implement generic applications that operate over the complete data space [174]. From the technical perspective, linked data uses RDF as a standardized data representation format, and HTTP as a standardized access mechanism [175]. Different endpoints have been proposed to extract the data using the SPARQL query language.

Recently, considerable efforts [176] have been invested to bring the medical knowledge onto the Web using Semantic Web technologies [91] in order to integrate different sources and to share, update and reuse the knowledge. Noticeably, drug-drug interactions, drug-food interactions and adverse drug reactions are being published and accessible as linked data [177, 178]. For instance, Bio2RDF [16] offers different SPARQL endpoints to DrugBank[2] [12] describing the drug-drug and food-drug interactions. Some other efforts have focused on integrating multiple existing linked data into a common repository such as life linked data[3] which integrates 25 popular biomedical data sources.

Freshly, research works focus on integrating these knowledge sources in order to enrich their systems with external information. In this context, Ostankov et al.

[2]DrugBank SPARQL Endpoint: http://drugbank.bio2rdf.org/sparql
[3]Linked Life Data: http://linkedlifedata.com/sparql

[179] proposes a Linked Health Answers system for question answering. This system implements mechanisms that transform natural language questions into formal semantic request taking the form of RDF triple (<?s,?p,?o>) based on NLP tools and machine learning-based algorithm. Once the request is transformed, it will be invoked over the ontology connector (e.g. Linked Life Data, FreeBase and DBpedia). In the pharmaceutic industry, Jentzsch et al. [175] referred to LinkedCT, DrugBank and Diseasome repositories in order to integrate accessible data related to companies, drugs, diseases and genetic variation, and continuously keep the user up-to-date with the available extra data.

Medical Open Linked Data seems to be promising for generating personalized decision. In this context, Khalili and Sedaghati [180] proposed the Pharmer system, an intelligent medical prescription system. Pharmer is a collaborative system that involves the patient, the physicians, the pharmacists and pharmaceutical researchers. It integrates various open linked data such as DrugBank, DailyMed and RxNorm in order to automatically detect the drugs and semantically annotate the e-prescription. It offers up-to-date information about the drugs coming from multiple dynamic data sources. However, the authors didn't detail how the system may help the physicians selecting the right treatment strategy. Moreover, they didn't evaluate the impact of integrating external linked data on the system performance.

5.2.4 Discussion

Automating the decision-making is crucial in healthcare to provide a support to health professionals for processing huge available medical knowledge combined with the patient context for a personalized treatment. But, it remains limited only to provide recommendations or alerts as always the final decision comes to the physician. In general, the discussed works refer to adaptation as the process of mapping the generic recommendations to the patient profile without considering the dynamicity of patient context. Most of them deal with the medical knowledge formalization in order to automate the decision-making. Mainly three techniques have been used: data mining [166], HTN planning [167, 168, 169] and ontology reasoning [165, 170, 171, 172, 173] which is the most used technique. However, based on our survey, we found that some important points need to be considered for the smart management of the patient treatment:

- Mechanisms for the adaptation of the patient treatment to accelerate the decision making and providing a preventive management of the patient to avoid health complications, except the work of Lasierra et al. [36] and Sánchez-Garzón et al. [169].
- Encoding the characteristics of the medical interventions in order to provide flexible schema that underpins the dynamic generation of the patient treat-

ment based on the context changes.

- Following a methodology to formalize the medical knowledge for chronic disease management and build the appropriate ontology based on the system requirements.
- Updating medical knowledge and decision rules. It is important to provide a collaborative user-friendly interface that allows not merely annotating the clinical guidelines, but also transferring the medical experts' tacit knowledge, who are not necessarily IT-experts.
- Integrating reusable external knowledge sources such as DrugBank to complete information within the clinical guidelines in order to provide personalized decisions and avoid medical errors. For instance, according to a study done on 4.152 diabetic patients [181], 62.3% had one or more medical errors.
- Evaluating the impact of the reasoning process on the system performance (response time and scalability), except the work of Lasierra [182] which focuses only on the monitoring and the analysis evaluation.

Consequently, we propose a prescriptive cognitive system that allows adapting the patient treatment through monitoring the patient vital signs, analyzing the captured data and planning for better personalized decisions in order to assist the physician in taking the right decision at the right time. In the next section, we provide an overview of the proposed system, mainly the patterns that we have instantiated, and we delineate the cognitive capabilities of the system for decision-making.

5.3 A Prescriptive Cognitive System for Personalization

In this section, first, we propose an overview of the reused patterns for the development of a prescriptive cognitive system. Then, we mainly deepen the procedural medical knowledge as well as the plan process for personalized decisions concerning the patient treatment.

5.3.1 System Architecture Overview

Following the model-driven methodology proposed in chapter 3, we combined a set of design patterns to provide a scalable prescriptive cognitive system for the smart management of the patient treatment. The design patterns that have been instantiated are the following:

- *The Prescriptive Cognitive Management Pattern*: we propose to instantiate this pattern to coordinate the monitoring, the analysis and the plan when

managing patients with comorbidity, while notifying with the physician with the generated plan. Thus, a set of management processes and their interactions are defined as well as the knowledge components.

- *The Semantic Knowledge Mediator Pattern.*: we propose to instantiate this pattern when acquiring the *SensoryKnowledge*, the *ContextKnowledge* and the *ProceduralKnowlegde* in order to enable the collaboration among the machines (wearables), the management processes and the human (experts); and to ensure knowledge sharing and reuse. Within this pattern, two ontological models are implemented the *Wearable Healthcare Ontology* describing both the SensoryKnowledge and ContextKnowledge (presented in chapter 4, section 4.4.2); and the *Treatment Plan Ontology* describing both the ContextKnowledge and the ProceduralKnowlegde (presented in section 5.3.3). These two models are linked to each other through common concepts related to the context.
- *The Multi-tenant Management Process Pattern*: we propose to apply the Multi-tenant Management Process pattern in order to dynamically allocate IT resources to satisfy the system requirements and provide better performance. We have already highlighted the gain of applying this pattern when monitoring the system in chapter 4. In this chapter, we are mainly interested in evaluating the application of this pattern when deploying the procedural knowledge and the plan process.

We reused the *Cognitive Monitoring System* proposed in chapter 3 in order to guarantee the scalability, the big data management and semantic integration within the monitoring process. In this chapter, we enriched this system with a semantic platform, implementing the *Semantic Knowledge Mediator* pattern, in order to allow the medical experts populating their medical knowledge (procedural knowledge) to be automatically reused by the plan process for decision-making. It is worth mentioning that we consider that the execution of the analysis process is done by the experts. As previously mentioned in our model-driven methodology, the development of smart IoT-based systems requires formalizing the procedural knowledge, called also tacit knowledge, based on the collaboration of domain experts (in our case, the medical experts). Thus, the next section provides an overview of the proposed methodology.

5.3.2 A Collaborative Methodology for Medical Knowledge Formalization

Medical knowledge formalization is a complex task, and requires a deep understanding of the domain and of the disease management strategies for the treatment

personalization. Thus, medical experts' collaboration and participation is required when building the ontology to provide an efficient knowledge transfer and sharing. Nevertheless, some cultural and social factors may hinder the progress of this process, especially that participants may feel and think that knowledge sharing depletes the time and the efforts that can be invested in other activities more beneficial for themselves [183]. Consequently, we propose a collaborative methodology for the tacit knowledge capture. To deal with the previously stated issues, we defined three phases: the *preparation*, the *knowledge organization* and the *knowledge reuse*.

The *preparation phase* is introduced to lighten the issue pertaining to the availability of medical experts. Hence, despite directly contacting them, the knowledge engineer proceeded with studying and collecting from the literature information about treatment personalization, and identifying techniques and tools related to the medical knowledge representation (1). Besides, the knowledge engineer reviews existing patient records and use cases (1') to extract the context, and review also the clinical guidelines and consensus for personalization to identify the diseases management strategies (1"). During this phase, an iterative process is identified to refine the identified concepts (2). The output of this phase is the identification of patient related information such as the preferences, the personalized goal, history, symptoms, etc., as well as information related to medical knowledge such as the characteristics of medical interventions including contraindications, side effects, interactions, alternatives, parameters to be monitored, etc. (Figure 5.1). The collected information is the input to the second phase.

The second phase is the *knowledge organization*. It is based on the collaboration with medical experts to define the need for the personalization process and the key elements for the decision making. To avoid barriers related to social and psychological dimensions when collaborating with experts, our methodology adopts an incremental approach in which the identification of the knowledge structure is decomposed into sub-steps. Thus, we organized meeting with each expert to validate and enrich the defined concepts (3, 4). We collaborated with medical experts from different domains (General Doctor, General Surgeon, Endocrinologist, and Radiologist) in order to identify a generic representation that can be specialized and instantiated in different domains. By repeating the processes (3, 4), different models are generated. In this way, problems such as being influenced by each other as well as the difficulty to find a common available time slot for meeting are covered. Furthermore, according to the expert character, different methods and questions are adopted to extract his/her knowledge. For example, based on the preparation phase, the knowledge engineer animates the discussion in order to deal with problems pertaining to people less in agreeableness or less in extraversion.

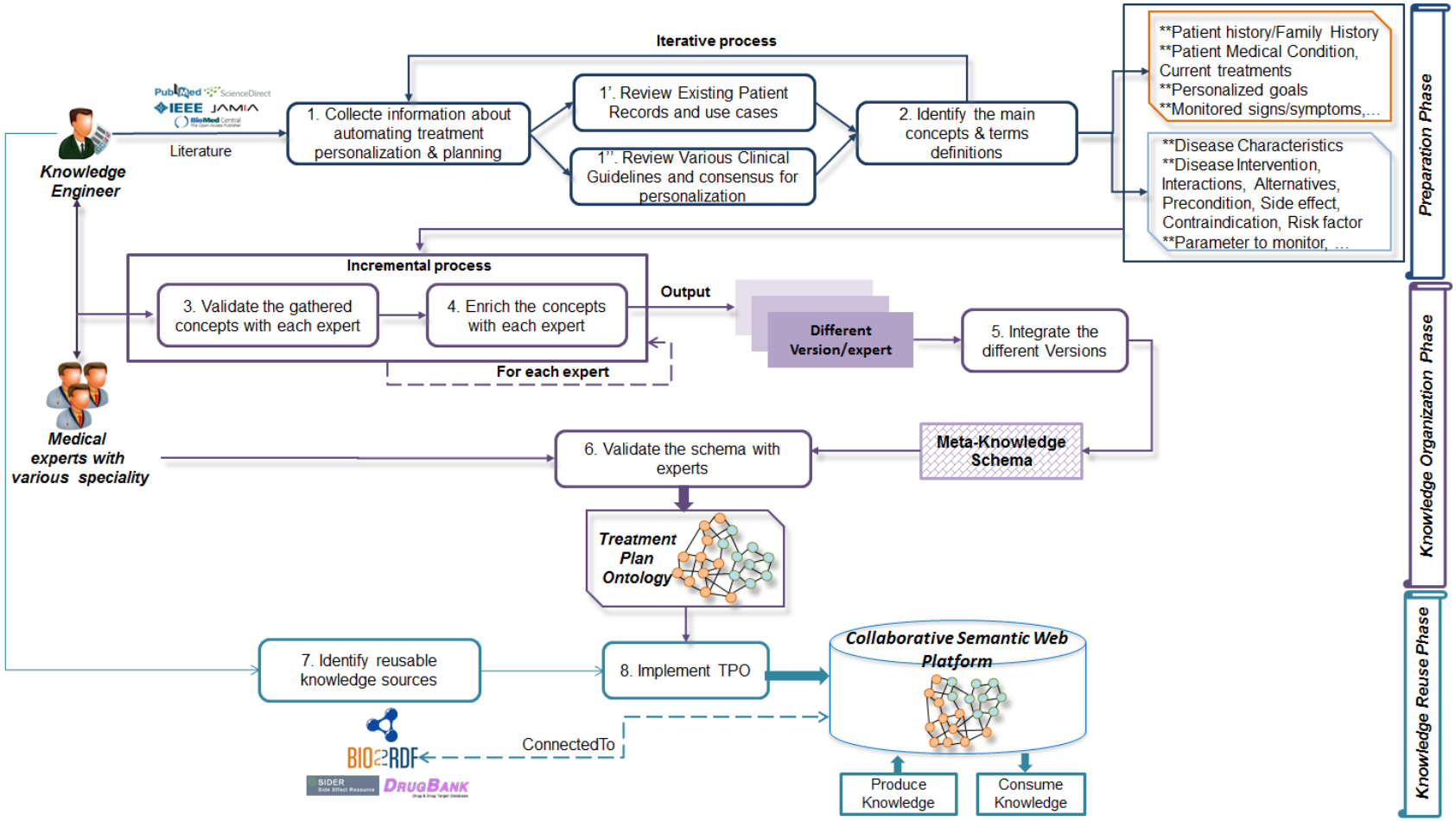

Figure 5.1: A three-phase methodology for the Treatment Plan Ontology design and implementation

The next step consists in integrating the generated models (5) to provide a unified ontology schema which is approved and validated by the experts (6). When integrating the different models, we tried to provide a trade-off between the expressiveness of the representation language and the difficulty of the reasoning over its representation. Indeed, the more expressive is the language, the harder is the reasoning [184]. The output of the knowledge organization phase is the "*Treatment Plan Ontology*" (TPO) model that depicts a common and coherent representation of the chronic disease treatments based on ontology. TPO will be the basis for knowledge acquisition and reasoning.

The final phase is the *knowledge reuse* phase. It includes the implementation of TPO in a computer interpretable format (8) in order to produce and consume the knowledge. But before implementing TPO, the knowledge engineer identifies existing reliable medical knowledge sources to be reused (7) such as DrugBank published as linked data.

Similar to existing methodologies for building ontologies, detailed in (chapter 2, section 2.4.1.3), our proposed methodology relies on common steps for the ontology construction (such as the identification of requirements, the identification of the ontology terms and the reuse of exiting ontologies), and extends them with a well-defined workflow that involves experts in the conceptualization and schema organization phase. Contrarily to existing methodologies, our methodology takes into consideration social and psychological factors that impede capturing the experts' tacit knowledge, and the collaboration process when constructing the meta-knowledge.

To conclude, our methodology relies at a first stage on face-to-face exchange for the identification of the meta-knowledge reflecting experts' vision based on a formal representation. Then, the knowledge is collaboratively populated in a sharable virtual environment through a user-friendly interface. It is worth mentioning that our methodology has been generalized in order to be adopted in other domains such as scientific research activities management [185]. In the next section, we deepen the TPO structure, and we evaluate its capabilities to provide personalized treatment pertaining to a specific patient having the hyperglycemia in type 2 diabetes.

5.3.3 Treatment Plan Ontology

We propose a flexible TPO schema that builds the bridge between the artificial intelligence planning and the semantic representation to grasp the computational intelligence. Planning is defined as "*the reasoning side of acting. It is [...] process that chooses and organizes actions by anticipating their expected outcomes*" [186]. Thus, the proposed TPO [187] semantically represents the medical interventions as actions, characterized through preconditions and side effects, that aim at reaching the patient objective. The ultimate purpose is to provide a generic model that fos-

ters automating the decision making according to the detected problems. For clarity reasons, Figure 5.2 portrays the main classes and relationships of our TPO formalizing a problem-solving knowledge for chronic disease management. Referring to the knowledge pattern (section 3.4.2.1), TPO portrays the *ProceduralKnowledge* describing the medical interventions to manage chronic diseases (orange color); and the *ContextKnowledge* describing the patient conditions (yellow color). An ontological representation of TPO using protégé is presented in Figure A.3.

The *ProceduralKnowledge* within TPO introduces the medical condition class which is defined by the Segen's Medical Dictionary as "*A disease, illness or injury; any physiologic, mental or psychological condition or disorder ... A biological or psychological state which is within the range of normal human variation is not a medical condition*". In our work, we consider measureable conditions which are identified through measurable parameters, for instance, the HbA1c identifies a high or low blood sugar medical condition. We consider that a disease has symptoms which are, in turn, medical conditions. Idem, each medical condition has risk factors which are defined as conditions that could make a person more likely to develop a disease or to amplify the symptoms of an existing disease. For example, both obesity and hypertension are risk factors of the type 2 diabetes. These medical conditions can be minimized and/or stabilized through medical interventions that help achieving the patient goal. An intervention can be behavioral, surgical or drug-based characterized by its route of administration.

To automate the selection of a treatment, our TPO associates each medical intervention with the appropriate disease strategy, and encodes the strategies hierarchy through the HasSuccessorStrategy property. For example, in type 2 diabetes, four strategies are identified: the "monotherapy", the "dual combination", the "triple combination" and the "complex insulin" [188]. If a patient, who is following a "monotherapy" strategy, presents an increased blood sugar, it is recommended to prescribe for him/her a "dual combination" strategy. Other properties have been introduced such as HasPrecondition and HasIntervPrecondition to respectively verify criteria related to the patient conditions, and encode the disease interventions ordering like prescribing the "sulfonylurea" if the "metformin" fails. Moreover, to provide personalized treatments, TPO describes the medical contraindication as well as the interactions and side effects expressed as conditions that foster the integration of external reliable knowledge sources such as DrugBank.

Focusing on treatment personalization, the *ContextKnowledge* within TPO includes concepts related to the patient profile such as diseases, medical conditions including the allergies, the monitored signs, the family history, the current treatments and preferences. Likewise, each patient has a specific TargetGoal, which is already represented in WH_O and used by the *Analysis* process to detect personalized anomalies. Each anomaly is associated to a medical condition in order to be explored by the *Plan* process to extract the interventions addressing it.

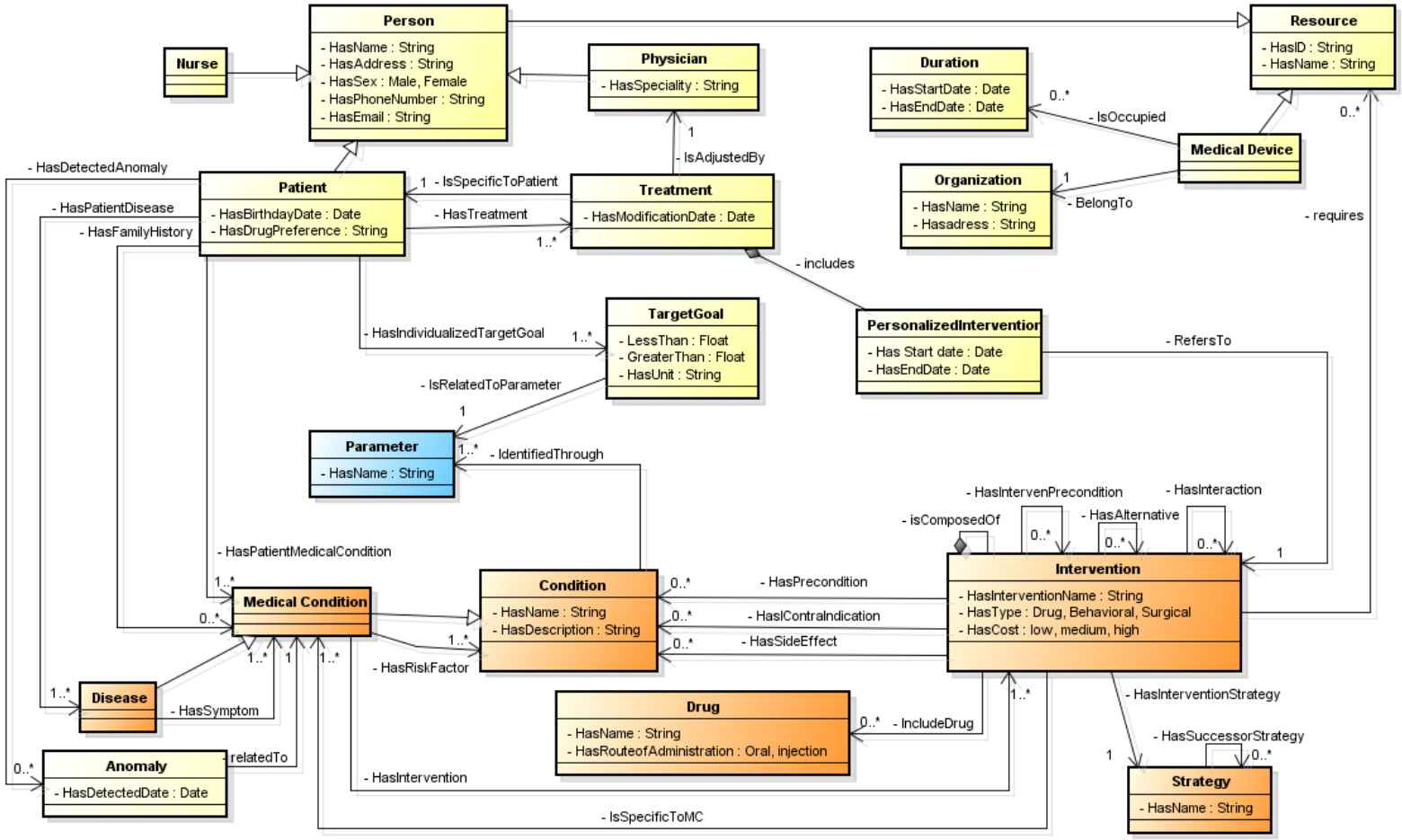

Figure 5.2: Depiction of the main classes and relationships with their cardinalities in the proposed Treatment Plan Ontology

To conclude, TPO is a generic representation that semantically formalizes the medical interventions and patient context. It has been conceived to allow the medical experts populating the decision rules through creating relations among TPO instances, as well as to integrate existing knowledge sources. Thus, medical experts can easily update medical decision rules, which will be later reused by the management processes. To this end, we propose in the next section a collaborative semantic web platform that implements TPO to foster the development of the knowledge reuse phase.

5.3.4 A Collaborative Semantic Web Platform for Medical Knowledge Acquisition

One of the main challenges of acquiring knowledge is interacting with experts who are not necessarily IT-experts. To hide the complexity of annotating the medical interventions based on TPO, we developed a collaborative user-friendly platform that allows medical experts transferring and sharing their medical knowledge. Similar to the collaborative platform that we have developed for the annotation of the wearable characteristics in chapter 4, the medical knowledge acquisition platform is an instantiation of the *Semantic Knowledge Mediator* pattern, and it is based on the same semantic web technologies (Semantic MediaWiki and Apache Fuseki). We installed around 30 extensions to guarantee an easy interaction with the experts. Our collaborative semantic web platform integrates OpenLDAP[4] to control the access to the patient medical conditions. The aforementioned components are deployed in virtual containers hosted in Proxmox as presented in Figure 5.3.

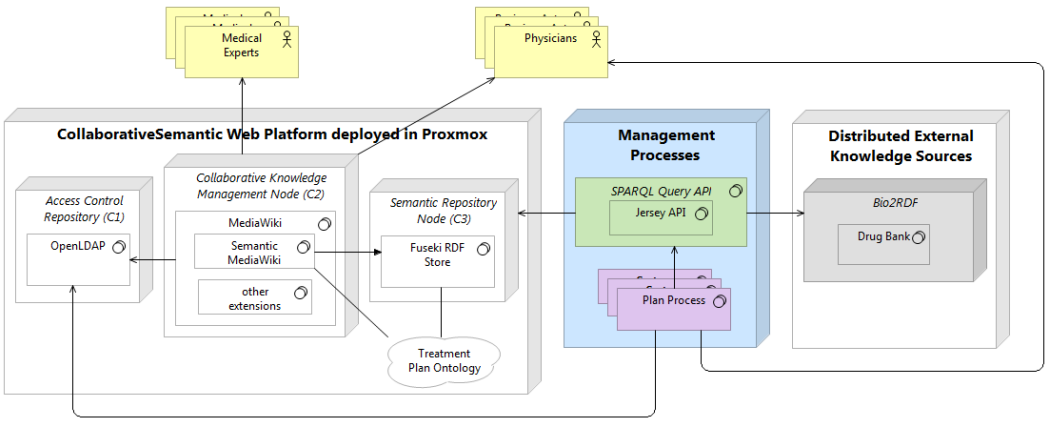

Figure 5.3: The main component of the proposed collaborative semantic web platform

[4]OpenLDAP: http://www.openldap.org/

The management processes consume the populated knowledge based on flexible SPARQL queries enacted through restful web services using the Jersey API to retrieve the required information from Fuseki. The elaborated SPARQL queries are generic and rely on the TPO schema. They do not integrate specific conditions related to the patient, which make them reusable.

In this thesis, we decided to instantiate our solution for hyperglycemia in type 2 diabetes in order to evaluate its usability by experts and its benefits in managing the patient treatment. Thus, the first step consists in annotating the medical interventions and strategies published by the American Diabetes Association (ADA) and European Association for the Study of Diabetes (EASD) [188] based on TPO. Figure 5.4 depicts a snapshot of user-interfaces used to annotate simple and composite medical interventions based on TPO. The user interface is personalized according to the selected item. For instance, if the selected strategy is "monotherapy", the list of drugs is displayed; else the list of interventions is displayed. In case of composite intervention, the list of drugs is automatically deduced from the simple interventions composing it based on inference techniques. Once these annotations are saved in the platform, they will be automatically stored in Fuseki and exposed as linked data. These annotations will be reused by the management processes and integrated with external knowledge sources for fine-grained decision.

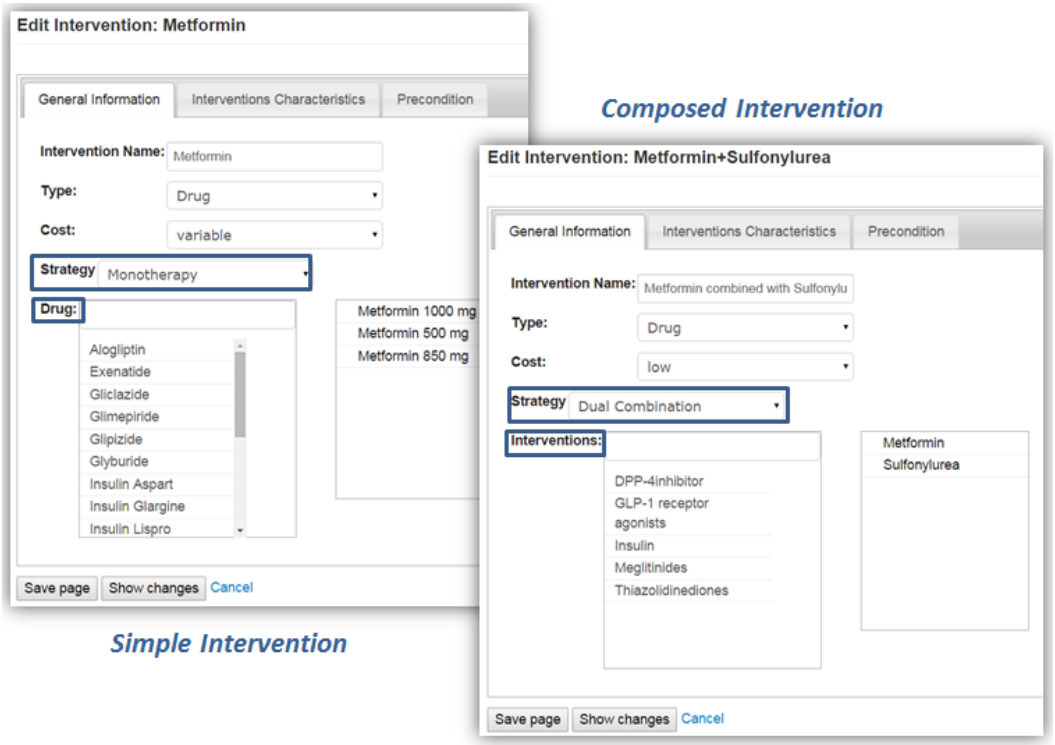

Figure 5.4: Semantic MediaWiki for medical intervention annotation

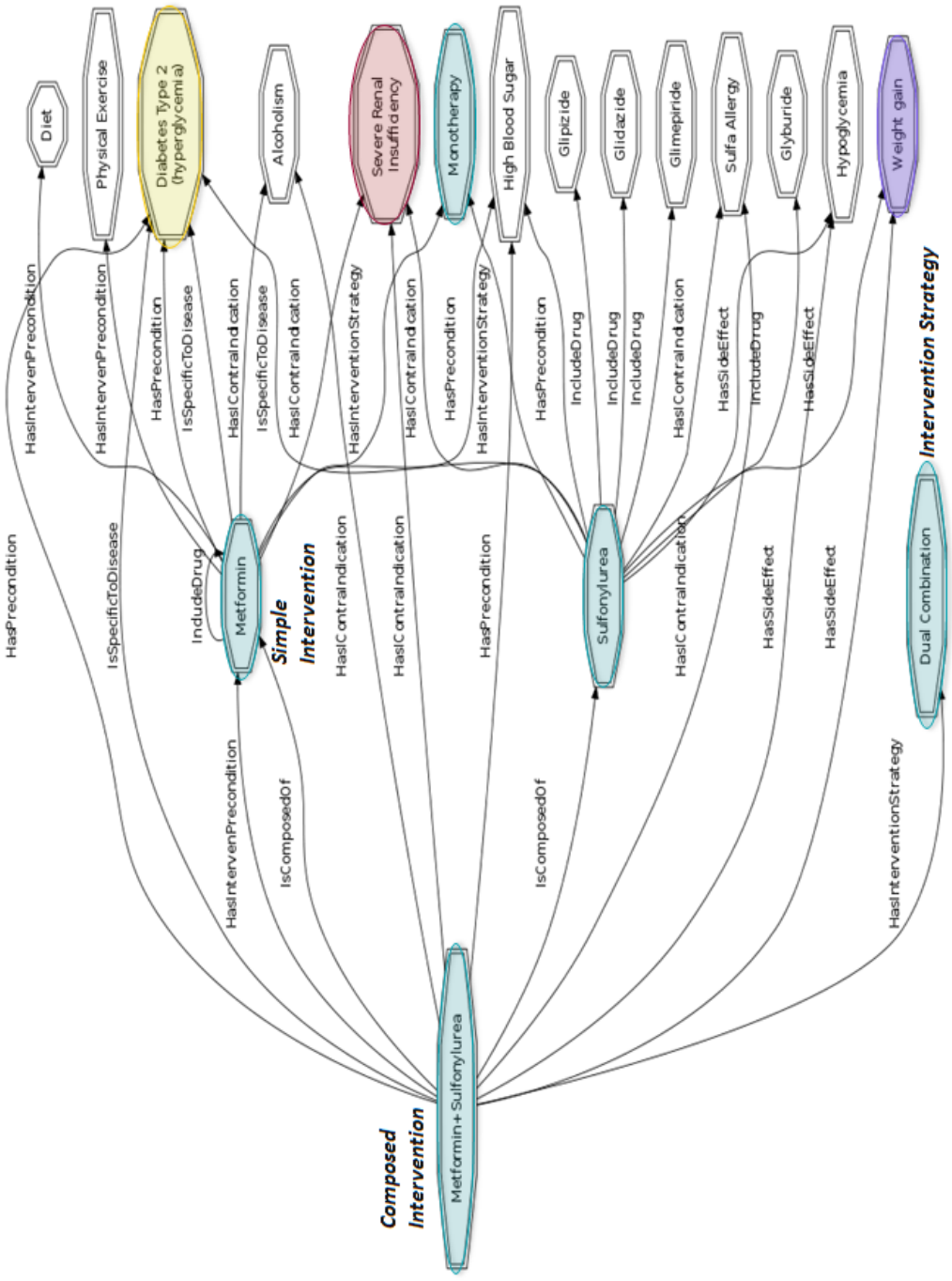

Figure 5.5: An expert of TPO instantiation describing diabetes treatments

Figure 5.5 portrays an excerpt extracted from the platform for the annotation of "Metformin combined with sulfonylurea" which is specific for managing "hyperglycemia". The shapes represent the instances of TPO classes, while the arrows represent the relations among these instances. This intervention is ascribed to the "Dual Combination" strategy, and it is composed of two simple interventions: the "Metformin" and the "Sulfonylurea". According to [188], this composed intervention is selected if the "Metformin" fails to manage the patient blood sugar. This statement is designed through the "HasIntervenPrecondition" property. The contraindications of the "Metformin combined with sulfonylurea" are deduced from the simple interventions composing it. For instance, the "Sulfa Allergy", "Severe renal insufficiency" and "Alcoholism" are contraindications deduced from the "Sulfonylurea" intervention. Likewise, the list of drugs of this intervention as well as their side effect are automatically deduced from its composite interventions (encoded from [188]: Figure 2 page 1371 and Table 1, page 1368).

In the next section, we delineate the planning algorithm that we propose to reason on TPO and DrugBank to automate the personalization of the patient treatment.

5.3.5 Ontology-based Planning Algorithm for Treatment Personalization

A patient-centered approach refers to "*providing care that is respectful of and responsive to individual patient preferences, needs, and values, and ensuring that patient values guide all clinical decisions*" [188, 189]. In this context, we propose a planning algorithm that meets the individual needs of each patient, while avoiding contraindications and drugs interactions as well as minimizing side effects of the medical interventions and the risk factors of developing new diseases. Figure 5.6 portrays the main steps for the generation of personalized recommendation for the chronic disease management. The proposed workflow has been validated with the medical expert.

Algorithm 1 represents a formalization of the proposed decision workflow that operates on TPO and DrugBank instances. The algorithm is generic and does not hardcode the medical decision rules and disease interventions. It takes as input: the patient identifier (idp_i), the list of detected anomalies sent by the *Analysis* process, the TPO individuals and relations describing the medical intervention and medical conditions, and the DrugBank knowledge base [12] describing medical interactions. The first step (Step 1) consists in initializing the parameters and extracting the patient conditions (PC) including the medical conditions (MC), and current treatments (PI). Then, the proposed planning algorithm proceeds with selecting, among the possible interventions corresponding to the right strategy (I), the interventions that address the patient detected anomalies and do not contraindicate the patient medical conditions (Step 2). Once the first list of possible recommendations

is established, the algorithm filters these recommendations by eliminating the interventions that interact with the patient current treatments, in case of comorbidity (Step 3).

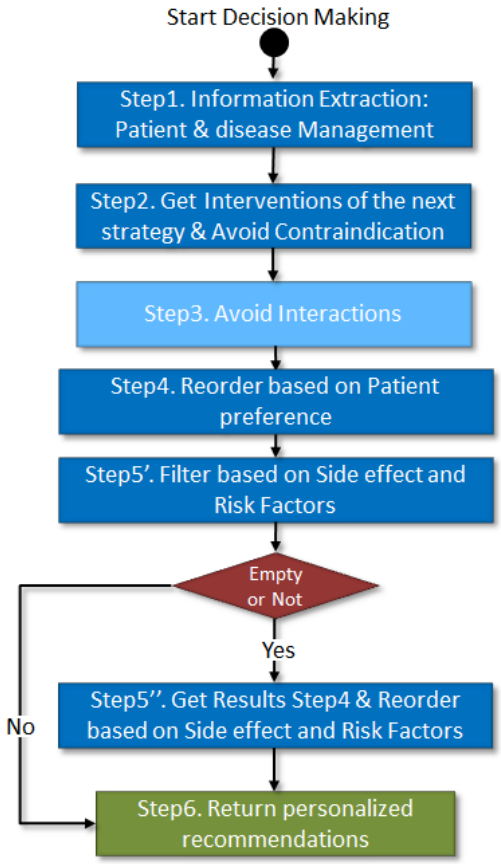

Figure 5.6: The workflow for personalizing treatment decision

At this stage, the DrugBank knowledge base is reused to detect drug interactions. After that, the planning reorders the generated personalized recommendations according to the patient preferences (Step 4). Indeed, it is important to take into consideration the patient preferences when elaborating the treatment [190]. The next step (Step 5') consists in eliminating the interventions those side effects may amplify the patient medical conditions or diseases risk factors. If this step returns recommendations, they will be automatically displayed (Step 6); else the planning goes to Step 5" to reorder the interventions generated in Step 4 based on the side effects and the disease risk factors. Finally, the algorithm consists in returning the list of the personalized recommendations (Step 6) to the physician who will select or propose the appropriate intervention.

Algorithm 1 An ontology-based planning algorithm for patient treatment personalization

Input: idp_i: the identifier of patient i; *A= {a_{1i},..., a_{ni}}* /* the detected anomalies of patient idp_i */
Data: Ontology functions (TPO individuals and relations); DrugBank Knowledge Base
Output: the list of the ranked personalized interventions

/***Step1. Initialization: Information Extraction***/
PC←GetPatientCondition(idp_i) /**PC= {pc_1,..., pc_m} m≥1*/
MC←GetPatientMedicalCondition(idp_i) /**MC= {mc_1,..., mc_l} l≥1*/
PI←GetPatientCurrentTreatment(idp_i)/**PI= {pi_1,..., pi_t} t≥1*/
Strategy←GetPatientNextStrategy(idp_i)/*Get the Next Strategy*/
I←ExtractInterventions(disease, Strategy) /**I= {I_1,..., I_v} v≥1*/
/***Step2. Extract interventions addressing the detected anomalies and belonging to the next strategy while avoiding contraindications***/
SelectedInterv← empty_array;
h=0;
for $a \in A$ **do**
  ∀ 1≤j≤v, 1≤k≤l
    if *((relatedTo(a, mc_k)≠ ∅) ∧ (IsSpecificToMC(I_j, mc_k)≠ ∅) ∧ (HasPrecondition(I_j) ⊆ PC) ∧ ((HasContraIndication(I_j)∩PC = ∅))* **then**
      SelectedInterv[h]← I_j;
      h=h+1;
  end
end
/***Step3. Avoid Interactions by referring to DrugBank Knowledge Source***/
for $st \in SelectedInterv$ **do**
  ∀ 1≤k≤size(PI)
    if *(dr:ddi-interactor-in(st, pi_k)≠ ∅))* **then**
      remove *st* from *SelectedInterv*;
  end
end
/***Step4. Ranking the interventions according to patient preference***/
size=length(SelectedInterv)-1;
x=0;
PrefRankedInterventions←empty_array;
for $p \in PersonalizedInterv$ **do**
  /*Extract the list of drugs included in the intervention p*/
    D←IncludeDrug(p);
    if ∄*d∈D, HasRouteofAdministration(d)∩HasDrugPreference(idp_i)=*∅ **then**
      PrefRankedInterventions[size]←p;
      size=size-1;
  else
      PrefRankedInterventions[x]←p;
      x=x+1;
  end
end
/***Step5'&5". Filter the interventions those side effects amplify the patient condition or their risk factors***/
PersonalizedInterventions ←empty_array;
RankedInterventionsWithSideEffects ←empty_array;
n=0;
b=0;
for $st \in PrefRankedInterventions$ **do**
  if *((HasSideEffect(st)∩PC≠ ∅) ∨ (HasSideEffect(st)∩HasRiskFactor(mc_k)≠ ∅))* **then**
      RankedInterventionsWithSideEffects[b]←st;
      b=b+1;
  else
      PersonalizedInterventions[n]←st;
      n=n+1;
  end
end
/***Step6. The final personalized recommendations***/
if *PersonalizedInterventions≠ ∅* **then**
  return ($PersonalizedInterventions$);
else
  return ($RankedInterventionsWithSideEffects$);
end

5.4 Clinical Evaluation

During the evaluation, we collaborated with an endocrinologist from the "Hedi Chaker Hospital" in Sfax-Tunisia to validate the efficiency of the proposed TPO and the planning algorithm. We identified with the medical expert two use cases and we referred to the literature to illustrate the algorithm behavior. For each use case, we provide a description of the patient profile, we detail the execution of the proposed planning algorithm and finally compare what the system generates to the medical expert's advice. It is worth noting that the list of drugs annotated in the semantic platform are extracted from [188] Table 1, page 1368.

5.4.1 Use Case 1

Patient Profile. The use case describes a 61 years old women. She has had diabetes since 2008. The patient is not obese and no family history has been reported. Her medical history includes hypertension treated with Valsartan and hypercholesterolemia treated with Atorvastatin. On physical exam, her weight was equal to 67 kg and her BMI was 23.6 kg/m^2. Her first HbA1C was equal to 6.7% and clearance was equal to 60ml/min. She does not present any complication from her diabetes and does not have any preference. The patient HbA1c target goal is fixed by the physician to 7%. As a first treatment, the physician prescribed "*Metformin combined with sulfonylurea*".

During taking the following treatments, the next measurement of the HbA1C reveals 7.9%, which is greater than the patient goal, and the measurement of the clearance is equal to 25ml/min; which reveals a severe renal impairment. Thus, an adaptation of the patient treatment is required.

Simulation of the Planning Process. The patient is taking "Metformin combined with sulfonylurea" which is a dual combination. According to the encoded medical knowledge, the next strategy can be "dual combination" by modifying the drugs combination, "triple combination" by adding a third drug or "Insulin Injection". As the patient has severe renal insufficiency (clearance < 30ml/min), the drug-based interventions are systematically avoided. This rule has been encoded in TPO through considering "severe renal insufficiency" as contraindication of the diabetes drug-based interventions (see Figure 5.5). Moreover, as presented in Figure 5.7, no drug interaction has been detected. Thus, the possible interventions are "One Insulin" or "Multiple Insulin Daily Dose". When discussing with the medical expert, she claims that the patient was treated with "Multiple Insulin Daily Dose", and validates the reasoning of the planning algorithm.

```
---------------------- Patient Information Extraction ---------
The list of the patient's Current treatments, except : Diabetes Type 2 (hyperglycemia)
*** Valsartan
*** Atorvastatin
The list of Intervention belonging to the next Startegy and do not contraindicate the patient conditions are :
*** Insulin
*** Multiple_Insuline_Daily_Dose
---------------------- Start Drug Interactions--------------------
No Interactions have been detected
---------------------- Start the reordering process based on the preferences, side effect and Risk Factors
The patient has no prefences
1. Insulin:
{Insulin_Aspart=, Insulin_Glargine=, Insulin_Lispro=}
2. Multiple_Insuline_Daily_Dose:
{Insulin_Aspart=, Insulin_Glargine=, Insulin_Lispro=}
```

Figure 5.7: The ordered personalized treatments for use case 1

5.4.2 Use Case 2

Patient Profile. A 42 years old woman has had type 2 diabetes for 7 years. The family history includes obesity and diabetes. Her medical and surgical history includes hypertension treated with Methyldopa (Aldomet) and dyslipidemia treated with fibrates (Gemfibrozil). Five years ago, she was diagnosed with phlebitis (Thrombose veineuse) treated with Acenocoumarol (sintrom). On physical exam, her weight is 115 kg, height is 1.52 m, and her BMI is 49.7 kg/m^2. Her blood pressure is 140/70 mm Hg. She has no acanthosis nigricans or skin tags on the neck. The patient prefers oral drug. Her blood sugar (fasting) was 3.5 g/l and HbA1c is equal to 9.5%. The HbA1C patient goal has been fixed by the physician to 7%.

The current diabetes treatment is based on Metformin 850 mg (three times daily) while following a Diet. During taking the following treatment, the HbA1c reaches 9% which is greater than the fixed objective. Thus, it is important to adapt the patient treatment.

Simulation of the Planning Process. The enactment of the proposed algorithm starts first with extracting the patient information like the medical conditions and current treatments. In this use case, the patient is taking a monotherapy intervention to treat diabetes. So, the algorithm selects the interventions that belong to the "Dual Combination" and do not contraindicate with the patient conditions (see Figure 5.8). After that, the algorithm proceeds with detecting and eliminating interactions that may take place with the patient current treatments. Three interactions have been detected. The "Metformin+TZD" intervention has been eliminated from the list of recommendation because both the "Pioglitazone" and the "Rosiglitazone", which are drugs of the Thiazolidinedione, interact with "Gemfibrozil". Likewise, the "Exenatide" drug is eliminated from the list of the proposed drugs corresponding to "GLP-1 receptor agonist" (the last recommendation). The ob-

tained interventions are reordered according to the patient preference, the intervention side effect and disease risk factors. The final result is portrayed in Figure 5.8. When discussing with the medical expert to evaluate the generated recommendations, the expert chooses "Metformin combined with sulfonylurea" and selects as drugs "Metformin" with "Glimepiride".

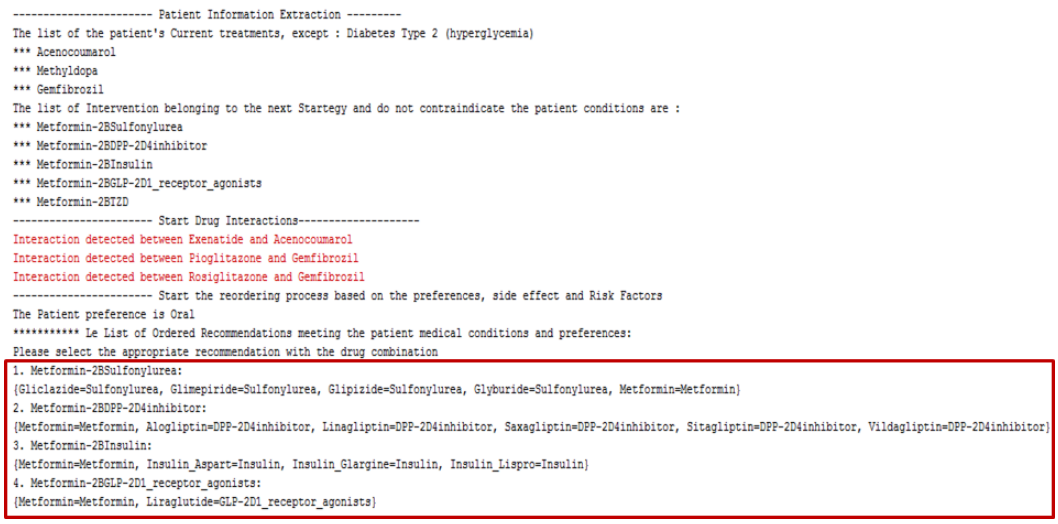

Figure 5.8: The ordered personalized treatments for use case 2

5.4.3 Use Case 3 from Clinical Diabetes Journal

We referred to a case study published in the Clinical Diabetes journal [191] in order to simulate the planning algorithm.

Patient Profile. A 67-year-old African-American man has had type 2 diabetes for 11 years. He was diagnosed incidentally through laboratory testing. Metformin was initiated at diagnosis and eventually titrated to his current dose of 1000 mg twice daily. The patient is self-referred to the clinic for help with blood glucose management. He checks his blood glucose once daily fasting. His medical and surgical history includes hypertension treated with lisinopril, hyperlipidemia treated with pravastatin, right-knee osteoarthritis, a right hip replacement at the age of 61 years, pneumothorax at the age of 35 years, and benign prostatic hypertrophy. He has no complications from his diabetes. On physical exam, his BMI is 31 kg/m^2. He has no acanthosis nigricans or skin tags on the neck. He has no peripheral neuropathy. His liver and kidney functions are normal, and urine microalbumin is negative. The HbA1C patient goal has been fixed by the physician to 7%. During taking the Metformin, the laboratory testing of the HbA1C (A1C) reveals 7.5%, which requires adjusting the patient treatment.

Simulation of the Planning Process. The enactment of the proposed algorithm is presented in Figure 5.9. The patient is following a "monotherapy" intervention, so the next strategy to be followed is the "Dual Combination". The algorithm extracts the list of interventions that belong to the "Dual Combination" and do not contraindicate the patient conditions. The next step focuses on avoiding the interventions that interact with the patient current treatment (Lisinopril and Pravastatin). In this use case, no interaction has been detected. So, the algorithm proceeds with ordering the selected interventions based on the patient preference (Oral). Then, it eliminates the interventions those side effects may amplify the patient medical conditions or emphasis the diabetes risk factors. The patient has obesity and hypertension history, so interventions that include the Thiazolidinediones (TZD) are avoided, because it may cause weight gain and heart failure. Moreover, the patient is at early stage of the hyperglycemia, thus, interventions that may cause hypoglycemia such as the Insulin and Sulfonylurea should be avoided.

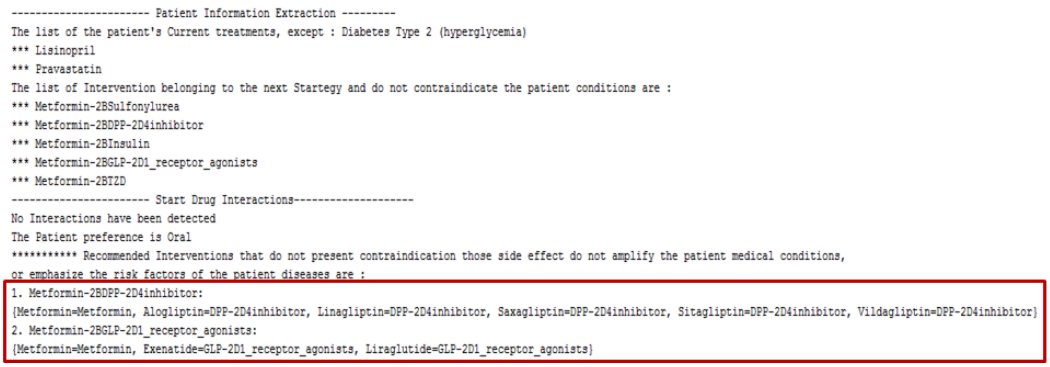

Figure 5.9: The ordered personalized treatments for use case 3

Hence, the algorithm retains two recommendations: (1) "*Metformin combined with DPP-4-inhibitor*" (oral drugs) and (2) "*Metformin combined with GLP-1 receptor agonists*" (oral+injection), which are ranked according to the patient preference (oral). Our algorithm proposes for each intervention the list of drugs which do not interact with the patient's current treatments. It is the role of the physician to select the combination of drugs. It is worth mentioning that the enactment of the proposed algorithm is limited to Step 5', since the algorithm is able to find the recommendations that fit the patient needs. The proposed recommendations are consistent with what the use case claims [191]. Moreover, we validated the result with the medical expert with who we collaborated in the aforementioned use cases. The medical expert claims that the first recommendation should be retained.

It is axiomatic that integrating DrugBank, which describes around 62,229 drug-

drug interactions[5] , within the decision algorithm is crucial to generate personalized decisions and avoid health complications. However, its integration leads to additional cost that may impact the system performance in terms of response time. In the next section, we evaluate the impact of integrating DrugBank into the planning algorithm.

5.5 Performance Evaluation

Despite the ability of generating personalized treatment, it is important to evaluate the planning performance in terms of response time and scalability management, especially when integrating external knowledge sources such as DrugBank. To this end, at a first stage, we simulated a set of scenarios that have been conducted on the following machine hardware configuration: memory 12GB; processor Intel(R) Core(TM) i7-2640M CPU@2.80GHz; and 64-bit Windows 7 operating system. We installed Proxmox on this machine and we created two virtual containers, hosting Ubuntu operating system. We attributed 6.5GB of memory to Proxmox distributed on the created virtual containers as follow: (Fuseki, 1 CPU, 2GB) and (Planning, 2CPU, 4GB). The Planning machine run the proposed planning algorithm which aggregates TPO instances (17,606 triples), hosted in Fuseki, with DrugBank which is hosted on an external server including around 1,376,989,061 triples[6].

We measured the response time of each use case presented in section 5.4 through using two approaches: (1) Online DrugBank Connection and (2) DrugBank Caching. Within the first approach, the planning algorithm connects to the external server hosting DrugBank to extract at runtime the detected interactions. While in the second approach, we defined a cache, which is periodically updated, storing the drug-drug interaction. Figure 5.10 portrays the response time of the planning algorithm for each use case. We noted that the response time highly depends on the complexity of the patient profile and on the adopted approach when integrating DrugBank. The caching approach is more efficient and provides faster response compared to the online connection, due to the network throughput connection when detecting interactions. Moreover, one drawback within the online connection is related to the availability and accessibility to DrugBank. If the SPARQL endpoint is not available, the algorithm is not enacted.

Second, we evaluated the scalability of the system when processing concurrent requests. Thus, we created multiple profiles similar to the use case 2, which is complex and includes the detection of drug interactions. Then, we incrementally increase the number of the simultaneous requests in order to (1) evaluate the ability

[5]Extracted from http://drugbank.bio2rdf.org/sparql in October 12, 2016
[6]Result extracted in October 12, 2016

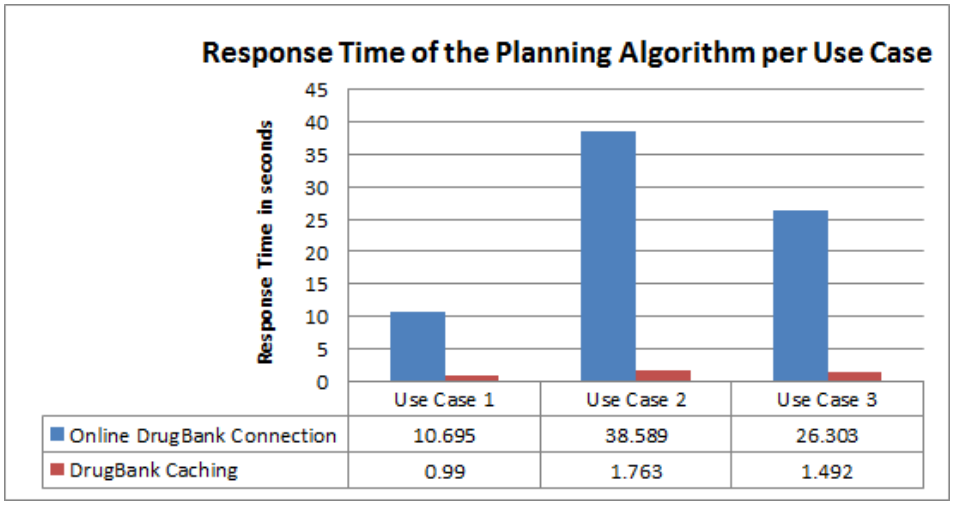

Figure 5.10: Evaluation of the planning response time per use case

of the system to process all requests and (2) and evaluate the average response time. Similar to the previous evaluation, we used the online and caching approaches to highlight the limitation of each one. Figure 5.11 portrays the system scalability and response time when managing simultaneous requests. We noted that the approach implementing the online connection to DrugBank is not scalable enough to support the execution of more than 40 requests in parallel, while the approach implementing a cache of the drug interactions may support the execution of 400 requests in parallel. From this evaluation, we retained the DrugBank caching approach for the planning and treatment personalization.

When running 400 requests in parallel, the average response time may reach around 4 minutes to successfully process all the incoming requests. This measurement is considered reasonable, when following a preventive approach for managing the patient treatment through instantiating the *Prescriptive Cognitive Management Pattern* and/or the *Autonomic Cognitive Management Pattern*. However, it is considered as a delay if the proposed planning algorithm is used by physicians during the consultation to get recommendations when adjusting the patient treatment.

We noted that, when running the experiments on a machine hosting proxmox (Planning, 2CPU, 4GB), while using the caching approach, the response time exponentially increases if the number of concurrent requests increases (due to IT resources constraint –memory and CPU). Thus, in complex systems where a high number of planning requests are sent, the planning may take a long time to respond, which is not recommended especially if we are managing patients with severe medical conditions and requires timely interventions. Moreover, despite using Proxmox to virtualize the IT resources and allow the users sharing the cost, the

system may fail processing all the incoming requests, if a big number of users are simultaneously demanding the service with IT resources constraints.

Indeed, the planning response time depends on the algorithm complexity, the implemented processing operations (used API and technologies), the complexity of the patient profile and the allocated IT resources (CPU and memory).

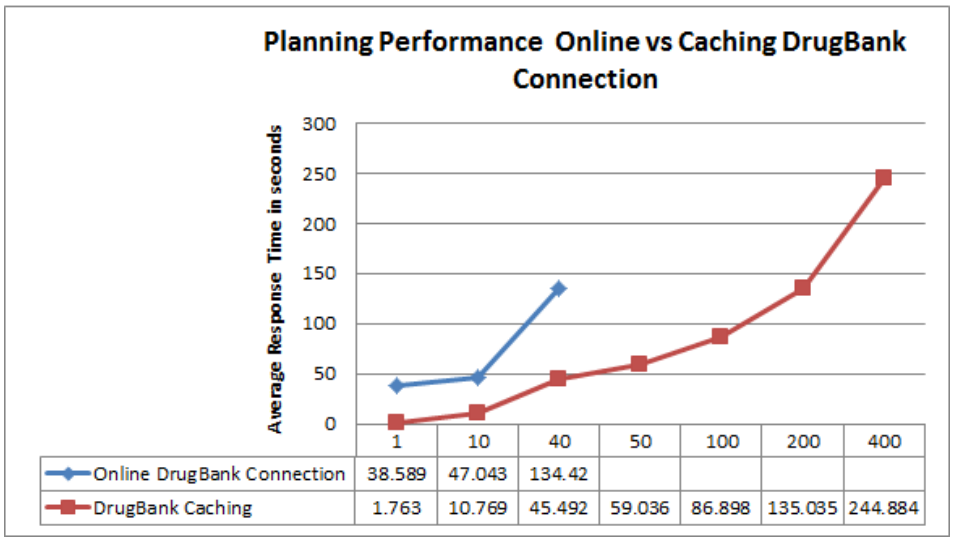

Figure 5.11: The impact of integrating DrugBank on system scalability & response time

To provide better performance and guarantee scalability, we propose to offer the planning algorithm as a service. Thus, we used the following powerful cloud server: 8 Intel(R) Xeon(R) CPU D-1521@ 2.40GHz, 128 GB of memory, and 1.77 TiB of disk storage. We run different experiments with different CPU configurations. Table 5.2 provides the response time of the planning service with different IT resources configuration. We hosted the same machines on a cloud server and we increased the memory to 20GB. We run again the same scenarios and we noted that increasing the memory potentially contributes to decreasing the response time compared to the first measurements as presented in Figure 5.12.

However, we found that the planning service slows down when receiving a high number of concurrent requests. For instance, within 2CPU, the average time for processing 4000 concurrent requests is around 18 minutes. Moreover, we noted that the 2CPU are fully used, and even the system presents congestion when managing the requests, which explains the delay when receiving the answers. Thereby, we varied the CPU configuration of the planning machine from 2CPU, to 4CPU and then to 6CPU. And, we observed the performance of the planning service regarding the response time. We noted that increasing the CPU limits drastically decrease the

response time. For instance, with 6CPU, the planning service is able to process the 4000 received requests 2.7 times faster than using only 2CPU on the cloud.

	Planning Response Time in second			
	on-premises	**on the cloud**		
Number of concurrent requests	Config (2CPU/4GB)	Config1 (2CPU/20GB)	Config2 (4CPU/20GB)	Config3 (6CPU/20GB)
1	1.763	1.457	1.214	0.671
10	10.769	6.502	3.694	2.525
40	45.492	19.08	10.307	6.468
50	59.036	24.514	12.098	8.279
100	86.898	35.443	19.793	13.436
200	135.035	69.192	31.918	23.532
350	219.391	110.149	59.003	29.536
400	244.884	129.39	66.007	45.036
500	294.765	156.565	79.835	54.444
1000	580.92	294.696	152.962	103.151
4000	2359.133	1111.086	566.076	400.956

Table 5.2: Planning performance evaluation based on the allocated IT resources

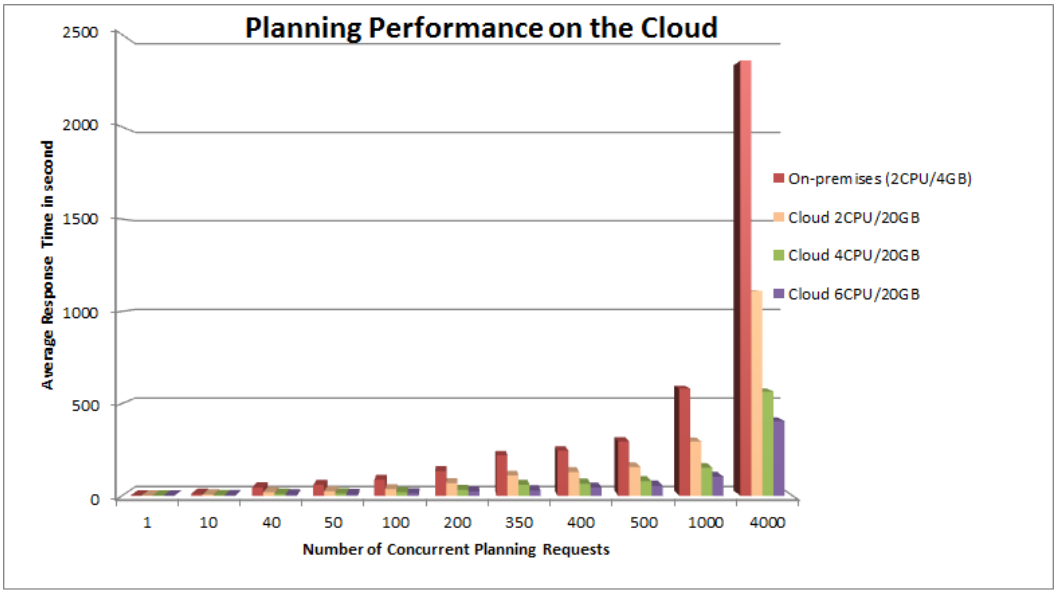

Figure 5.12: Impact of the CPU configuration on the planning performance

Consequently, finding a trade-off between IT cost and response time depends on the system requirements:

- If the planning service will be used within the *Prescriptive Cognitive Management Pattern* or the *Autonomic Cognitive Management Pattern* (running in the background) in order to prevent patient health complications, both Config1 and Config2 are acceptable, especially that the IT cost is shared with the users.

- If the system requires faster response, for example when managing patients with advanced stage of the disease, the Config 3 with 6CPU can be selected. In this case, the response time is decreased and the system is able to simultaneously serve 4000 change requests in around 7 minutes.

- If the planning service is intended to interact with the physicians at real-time, during the consultation for example, it is important in this case to provide faster response. So, in this case, the selection of the appropriate configuration depends on the number of the subscribed physicians to the service. For instance, if the number of physicians does not exceed 50, all users can share the same instance within the Config3. Thus, the response time does not exceed 9 seconds and the cost is shared among the users. In case of more complex system, where many physicians are subscribed, multiple instances of the planning should be offered as a service and shared by group of users according to the required response time.

5.6 Conclusion

To go towards a prescriptive cognitive management of the patient treatment, IoT-based systems should collaboratively interact with the domain experts in order to acquire the know-how, and in turn, assist them with the right decision at the right time based on cognitive capabilities.

Thus, in this chapter, we mainly concentrated in detailing the procedural knowledge and the plan process within the prescriptive cognitive management pattern for the adaptation and personalization of the patient treatment. Thus, we collaborated with medical experts to identify a flexible and generic ontological model, named TPO, describing the medical interventions for the management of chronic diseases. TPO has been instantiated and specialized for managing hyperglycemia in type 2 diabetes in order to evaluate its flexibility when reasoning to generate the right treatment based on the patient profile. We proposed an ontology-based planning algorithm that integrates TPO with DrugBank in order to generate personalized decisions. We demonstrated the efficiency of the proposed planning algorithm

through three real use cases that have been validated with the medical expert. Likewise, we evaluated its performance through measuring the average response time, and its scalability on the cloud. Finally, we provided some recommendations that help selecting the appropriate IT configuration based on the system requirements.

CHAPTER 6

Conclusion and Perspectives

"Perfection is achieved, not when there is nothing more to add, but when there is nothing left to take away."

- Antoine de Saint-Exupery

Contents

6.1 Conclusion 139

6.2 Perspectives 143

In this chapter, we firstly provide a summary of the main contributions and results accomplished during this PhD thesis. Then, we present a set of enhancements and give an insight into new research directions that enrich this work.

6.1 Conclusion

Nowadays, the Internet of Things (IoT) phenomenon is fast gaining momentum and touches different applicative domains. Its speed of adoption is creating challenges for IT professionals due to the increased number of connected things as well as their heterogeneity. Given, that these challenges impede the integration of IoT-based systems for better business benefits, this PhD thesis comes to provide new solutions that foster the development of smart IoT-based systems, particularly in the healthcare application domain. The achieved work is summarized below.

Autonomic Cognitive IoT Design Patterns

The sheer scale and design complexity of IoT-based systems lead us to propose a set of design patterns that drive the architect providing flexible IoT architecture. In this sense, chapter 3 defines a model-driven methodology for the design of smart IoT-based systems. The proposed methodology adopts different refinement levels that allows incrementally incorporating the system requirements through modeling the interactions among management processes and knowledge sources. The

proposed design patterns represent the core of our methodology. They have been proposed to deal with the system functional requirements through delineating the coordination of the management processes and their interactions with human; and the system non-functional requirements through ensuring the semantic integration, and managing big data and scalability issues. The proposed patterns are classified into three levels:

- *Management Processes' Coordination level*
  - **Knowledge pattern**: Inspired from the human memory model, this pattern organizes the knowledge for the smart management of IoT-based systems through composing it into sub-components: *SensoryKnowledge*, *ContextKnowledge* and *ProceduralKnowledge*.
  - **Cognitive Monitoring Management pattern**: This pattern enables the interaction of IoT systems with human. It identifies a bidirectional interaction: *IoT-Human interaction* to visualize the data, extract new insights and receive notifications in case of context changes; and the *Human-IoT interaction* to manage the system through modifying its context and allow the IoT-based system learning from experts and acquiring knowledge. Only the monitoring process is automated.
  - **Predictive Cognitive Management pattern**: It extends the Cognitive Monitoring Management pattern through modeling the coordination between the monitoring, the analysis processes and the expert. It also delineates the interaction of the management processes with the sensory and context knowledge in order to generate new knowledge about the managed element.
  - **Prescriptive Cognitive Management pattern**: This pattern coordinates the monitoring, the analysis and the plan processes, while interacting with the experts. It incorporates new mechanisms that allow the IoT-based system generating recommendations to assist the experts in taking business decisions.
  - **Autonomic Cognitive Management pattern**: To manage at runtime the system requirement evolution, this pattern proposes the dynamic discovery of management processes based on the system context. The dynamic discovery is achieved based on the *Management Process Ontology* which describes the characteristics of the processes and the conditions for their activation. This pattern can be combined with the aforementioned patterns in order to manage unforeseen requirements. The expert is always kept in the loop to validate the execution of the discovered processes and the generated plans.

- *Semantic Integration level*:
  - **Semantic Knowledge Mediator pattern**: This pattern proposes a solution to enable the collaboration among the machines (IoT devices), the human and the management processes based on semantic models describing the sensory, context and procedural knowledge. Thus, it guarantees the integrability as well as the knowledge sharing and reuse.
- *Big Data & Scalability Management level*
  - **Big Data Stream Detection pattern**: This pattern extends the *Cognitive Monitoring Management* to support data velocity, and specialize it for data stream processing. Thus, it delineates the monitoring process into sub-processes that understand and curate the received data, and retain attention.
  - **Big Data Analytic Predictive pattern**: This pattern extends the *Predictive Management* to support the big data management for batch processing. This pattern is also inspired from the human brain information processing: it imports data from external database into a temporary database (short-term memory), applies parallel batch-processing to harmonize the data and store it in clusters (long term memory). Then, this pattern introduces parallel data analytic service that may implement machine learning operating on the harmonized data clusters to generate new knowledge.
  - **Multi-tenant Management Process pattern**: This pattern is introduced to manage the system scalability and cost. Based on cloud computing principles, this pattern delineates the deployment of the management processes as well as the knowledge components. Thus, the tenants (i.e. consumers) may share or not the same processes with customized configuration, depending on the system requirements.

These patterns form the basis for designing and developing flexible smart IoT-based systems. The deployment of the proposed management processes and knowledge components requires a scalable platform that considers big data and data heterogeneity management. Thus, our second contribution focuses on offering a semantic cloud-based big data platform, named Knowledge as a Service.

Knowledge as a Service

Our KaaS extends the NIST cloud computing reference architecture with a new layer that aims at integrating heterogeneous data sources deployed on the cloud to generate new knowledge offered as a service. To deal with big data challenges,

our KaaS implements the NIST big data reference architecture identifying the following processes: data collection, data curation, analytics, visualization and access control. To manage the data heterogeneity, our proposed KaaS extends the NIST big data reference architecture with a semantic layer which includes the different knowledge components proposed in the Knowledge pattern. The proposed KaaS is generic and can be instantiated for any domain. In this thesis, we are interested in healthcare as an application domain, more precisely the patient health management. Thus, we proposed within the KaaS a *Cognitive Monitoring System*, which is the combination of the *Cognitive Monitoring Management* pattern, the *Big Data Stream Detection* pattern and the Semantic Knowledge Mediator pattern, for managing the patient health based on data stemming from wearable devices. So, domain ontologies describing the sensory, context and procedural knowledge are elaborated and integrated within the semantic layer. These semantic models are delineated in Chapter 4 and Chapter 5.

Wearable Healthcare Ontology

Chapter 4 proposed the *Wearable Healthcare Ontology* (WH_O) to foster the integration of heterogeneous data stemming from wearable devices. It is the foremost element which allows the proposed *Cognitive Monitoring System* understanding the meaning of the received data, as well as its structure. It reuses and extends IoT-O to support the wearable healthcare management as well as some concepts related to the patient context in order to generate personalized detection. Contrarily to existing works which store the observations in the ontology, our KaaS platform refers to the WH_O to understand and curate the observations, and then, store the harmonized data in big data clusters. In order to annotate the wearable devices based on WH_O, chapter 4 presented a collaborative semantic web platform based on Semantic Mediawiki. This platform represents an implementation of the *Semantic Knowledge Mediator* pattern. To evaluate the performance of the proposed cognitive monitoring system based on WH_O, we followed different KaaS configuration based on the *Multi-tenant Management Process* pattern in order to illustrate the ability of our KaaS to satisfy the system non-functional requirements.

Treatment Plan Ontology

Chapter 5 complements chapter 4 through enriching the *Cognitive Monitoring System* with new cognitive components and management processes in order to allow the IoT-based systems assisting the physicians through generating personalized treatment at the right time. This enhancement leads to the production of a *Prescriptive Cognitive System*, which combines the patterns used for the design of the Cognitive Monitoring System with the *Prescriptive Cognitive Management* pattern

to automate the decision making and the *Semantic Knowledge Mediator* pattern to describe the procedural knowledge (medical interventions). Thus, a semantic formalization of the medical interventions and their granular characteristics is presented through the TPO model in order to allow the IoT-based systems reasoning to automate the treatment personalization. To acquire the medical interventions, we elaborated a collaborative Semantic Web platform implementing the *Semantic Knowledge Mediator* to interact with the medical experts based on user-friendly interfaces. Furthermore, we proposed an ontology-based planning algorithm to integrate TPO with DrugBank for generating personalized recommendations. Following the *Multi-tenant Management Process* pattern, we evaluated the proposed planning service from the clinical and performance perspectives, and we provided a set of recommendations concerning the IT configuration according to the system requirements.

6.2 Perspectives

Research conducted during this PhD thesis helped addressing challenges which hinder the development of smart IoT-based systems. The outcomes of our research work open important and interesting research perspectives. Some of them are listed below.

- *Automate the Model Transformation*
  In this thesis, we proposed in chapter 3 a model-driven methodology for the design of smart IoT-based systems. Within this methodology, we defined a set of design patterns that satisfy both functional and non-functional requirements. We propose, as an extension of our work, the elaboration of a set of transformation rules that automatically refine the IoT-based system design based on the system requirements.

- *Optimization of the IT resources Allocation*

  In this thesis, during the evaluation phase, we illustrated the importance of providing a well-adapted configuration of IT resources and distributed architecture instance in order to cope with the system non-functional requirements. As extensions to the proposed KaaS platform, automatic configuration and reconfiguration decision models should be elaborated to dynamically manage the KaaS allocated IT resources based on the system requirements and context observations.

- *Automate Tacit Knowledge Acquisition*

  The collaborative methodology, which has been proposed in chapter 5 to extract the medical knowledge, aims mainly at formalizing the knowledge

structure and decision rules based on ontology to provide more flexible management. To acquire the medical knowledge, we have developed a semantic web platform offering user-friendly interface to facilitate the interaction with medical experts. To reduce the medical experts' efforts, we propose adding new cognitive features based on Natural Language Processing (NLP) techniques to extract the medical interventions from available text documents and annotate this knowledge based on TPO. Thus, the medical experts will validate/adjust the acquired knowledge before being used by the plan process.

- *Heuristic Approach for Personalizing the Patient Treatment*

  Current proposed planning algorithm doesn't consider the drug-doses when generating recommendations. Therefore, as an extension, we propose enriching the planning algorithm with machine learning and optimization capabilities in order to be able to calibrate the drug-doses. However, this perspective is constrained by the availability of medical databases describing the patients' profiles and their treatments.

APPENDIX A

Appendix

This appendix portrays the proposed ontologies using the ProtégéVOWL[1] plugin that offers user-oriented visualization of ontologies on protégé. The circle represents the ontology class, the arrow linking two circles represents the object property, and the green rectangle represents the data property of the class, while the yellow rectangle represents its type.

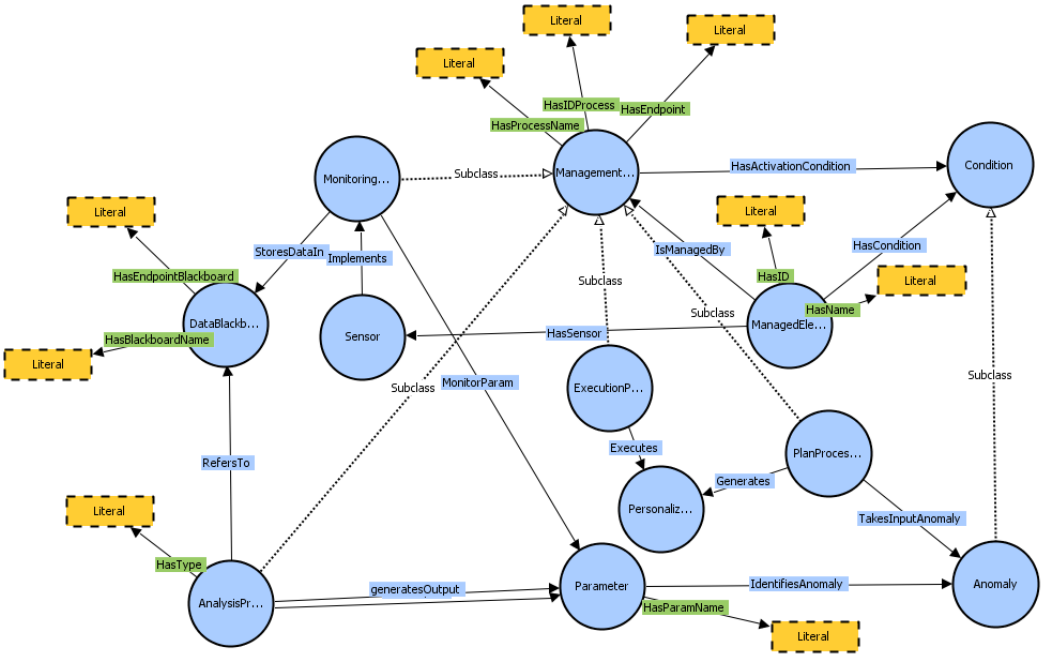

Figure A.1: MPO visualization using VOWL

[1] http://vowl.visualdataweb.org/protegevowl.html

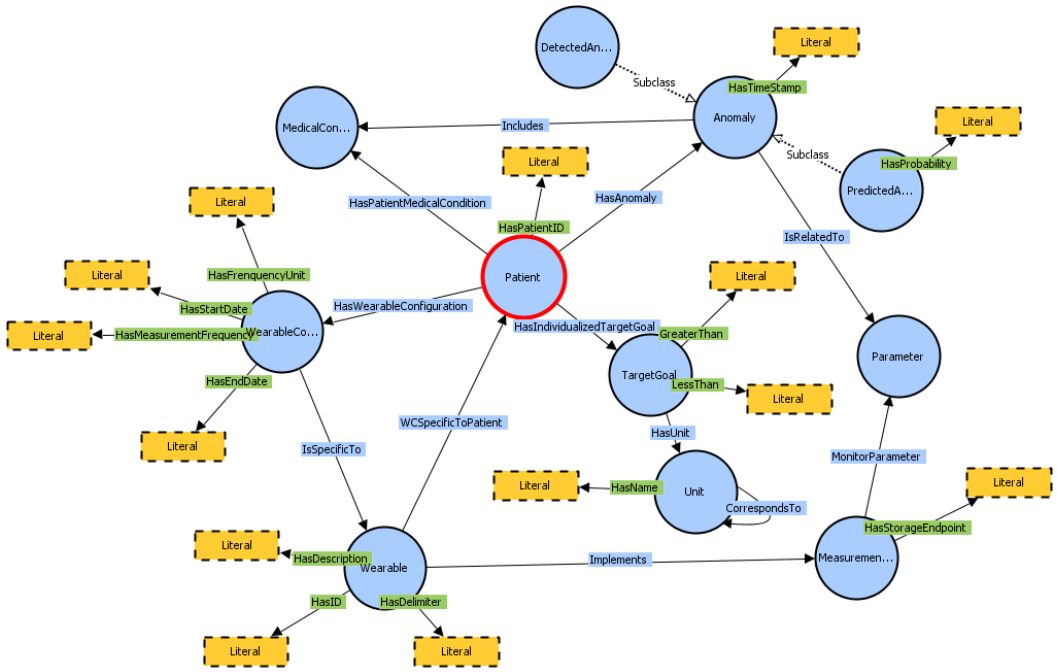

Figure A.2: WH_O visualization using VOWL

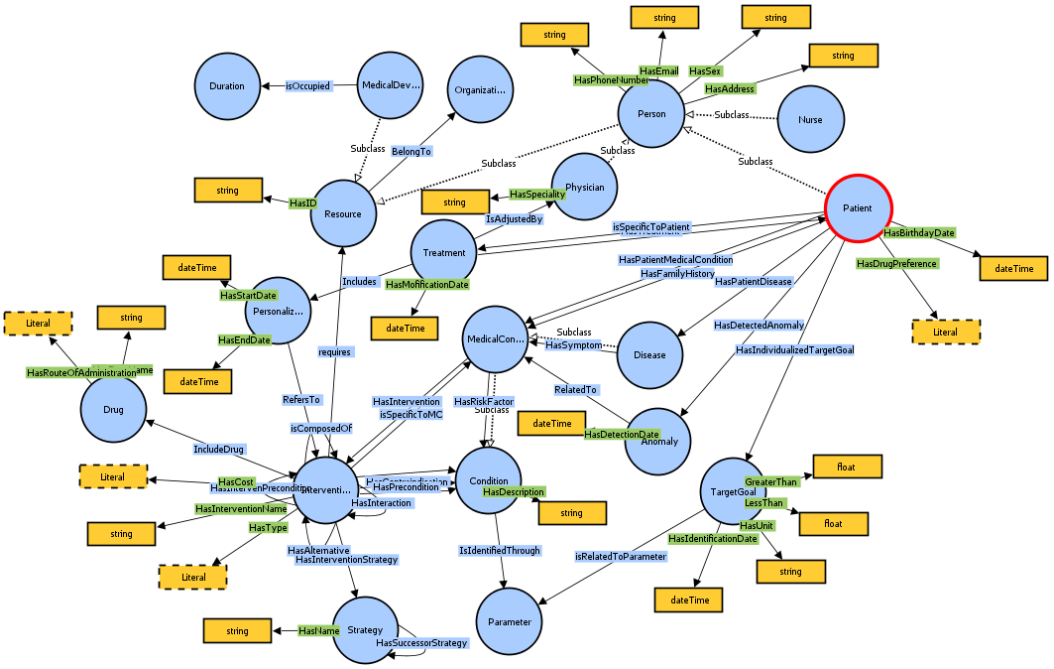

Figure A.3: TPO visualization using VOWL

References

[1] J. Nolte, "The human brain: an introduction to its functional anatomy," 2002. (Cited on page 1.)

[2] D. S. Bassett and M. S. Gazzaniga, "Understanding complexity in the human brain," *Trends in cognitive sciences*, vol. 15, no. 5, pp. 200–209, 2011. (Cited on page 1.)

[3] B. B. Biswal, M. Mennes, X.-N. Zuo, S. Gohel, C. Kelly, S. M. Smith, C. F. Beckmann, J. S. Adelstein, R. L. Buckner, S. Colcombe, *et al.*, "Toward discovery science of human brain function," *Proceedings of the National Academy of Sciences*, vol. 107, no. 10, pp. 4734–4739, 2010. (Cited on page 1.)

[4] Y. Wang, "Special issue on cognitive computing, on abstract intelligence," *International Journal of Software Science and Computational Intelligence*, vol. 1, no. 3, 2009. (Cited on pages 2 and 28.)

[5] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003. (Cited on pages 2, 8 and 17.)

[6] J. Kelly III and S. Hamm, *Smart Machines: IBM's Watson and the Era of Cognitive Computing*. Columbia University Press, 2013. (Cited on page 2.)

[7] J. E. Kelly, "Computing, cognition and the future of knowing," *Whitepaper, IBM Reseach*, 2015. (Cited on page 2.)

[8] Gartner, "Gartner Says 4.9 Billion Connected "Things" Will Be in Use in 2015." `http://www.gartner.com/newsroom/id/2905717`, 2014. [Online; accessed 15-September-2016]. (Cited on page 2.)

[9] A. Noronha, R. Moriarty, K. O?Connell, and N. Villa, "Attaining iot value: How to move from connecting things to capturing insights," 2014. (Cited on page 2.)

[10] V. Foteinos, D. Kelaidonis, G. Poulios, P. Vlacheas, V. Stavroulaki, and P. Demestichas, "Cognitive management for the internet of things: a framework for enabling autonomous applications," *IEEE Vehicular Technology Magazine*, vol. 8, no. 4, pp. 90–99, 2013. (Cited on page 2.)

[11] S. M. Meystre, G. K. Savova, K. C. Kipper-Schuler, J. F. Hurdle, *et al.*, "Extracting information from textual documents in the electronic health record: a review of recent research," *Yearb Med Inform*, vol. 35, pp. 128–44, 2008. (Cited on page 3.)

[12] D. S. Wishart, C. Knox, A. C. Guo, S. Shrivastava, M. Hassanali, P. Stothard, Z. Chang, and J. Woolsey, "Drugbank: a comprehensive resource for in silico drug discovery and exploration," *Nucleic acids research*, vol. 34, no. suppl 1, pp. D668–D672, 2006. (Cited on pages 3, 114 and 126.)

[13] M. A. Nicolelis, "Mind in motion," *Scientific American*, vol. 307, no. 3, pp. 58–63, 2012. (Cited on pages 3 and 83.)

[14] A. Gluhak, S. Krco, M. Nati, D. Pfisterer, N. Mitton, and T. Razafindralambo, "A survey on facilities for experimental internet of things research," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 58–67, 2011. (Cited on page 4.)

[15] R. De Lemos, H. Giese, H. A. Müller, M. Shaw, J. Andersson, M. Litoiu, B. Schmerl, G. Tamura, N. M. Villegas, T. Vogel, *et al.*, "Software engineering for self-adaptive systems: A second research roadmap," in *Software Engineering for Self-Adaptive Systems II*, pp. 1–32, Springer, 2013. (Cited on pages 4, 8, 19, 20 and 27.)

[16] F. Belleau, M.-A. Nolin, N. Tourigny, P. Rigault, and J. Morissette, "Bio2rdf: towards a mashup to build bioinformatics knowledge systems," *Journal of biomedical informatics*, vol. 41, no. 5, pp. 706–716, 2008. (Cited on pages 6 and 114.)

[17] J. Gantz and D. Reinsel, "Extracting value from chaos," *IDC iview*, vol. 1142, pp. 1–12, 2011. (Cited on pages 6 and 34.)

[18] H. Banaee, M. U. Ahmed, and A. Loutfi, "Data mining for wearable sensors in health monitoring systems: a review of recent trends and challenges," *Sensors*, vol. 13, no. 12, pp. 17472–17500, 2013. (Cited on pages 6, 9, 35 and 88.)

[19] S. Redmond, N. Lovell, G. Yang, A. Horsch, P. Lukowicz, L. Murrugarra, and M. Marschollek, "What does big data mean for wearable sensor systems?: Contribution of the imia wearable sensors in healthcare wg," *Yearbook of medical informatics*, vol. 9, no. 1, p. 135, 2014. (Cited on page 7.)

[20] D. Che, M. Safran, and Z. Peng, "From big data to big data mining: challenges, issues, and opportunities," in *International Conference on Database*

Systems for Advanced Applications, pp. 1–15, Springer, 2013. (Cited on pages 7 and 35.)

[21] E. D. U. with Research & Analysis by IDC, "The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things." `http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm`, 2014. [Online; accessed 15-September-2016]. (Cited on page 7.)

[22] M. B. Alaya, *Self-management, and Scalability for Machine-to-Machine Systems*. PhD thesis, INSA Toulouse, 2015. (Cited on page 8.)

[23] E. Mezghani, R. B. Halima, and K. Drira, "Draas: dynamically reconfigurable architecture for autonomic services," in *Web Services Foundations*, pp. 483–505, Springer, 2014. (Cited on page 8.)

[24] M. C. Huebscher and J. A. McCann, "A survey of autonomic computing–degrees, models, and applications," *ACM Computing Surveys (CSUR)*, vol. 40, no. 3, p. 7, 2008. (Cited on pages 8 and 19.)

[25] C. Klein, R. Schmid, C. Leuxner, W. Sitou, and B. Spanfelner, "A survey of context adaptation in autonomic computing," in *Fourth International Conference on Autonomic and Autonomous Systems (ICAS'08)*, pp. 106–111, IEEE, 2008. (Cited on page 8.)

[26] A. G. Ganek and T. A. Corbi, "The dawning of the autonomic computing era," *IBM systems Journal*, vol. 42, no. 1, pp. 5–18, 2003. (Cited on pages 8 and 18.)

[27] M. Ben Alaya, Y. Banouar, T. Monteil, C. Chassot, and K. Drira, "Om2m: Extensible etsi-compliant m2m service platform with self-configuration capability," *Procedia Computer Science*, vol. 32, pp. 1079–1086, 2014. (Cited on page 8.)

[28] D. Weyns, B. Schmerl, V. Grassi, S. Malek, R. Mirandola, C. Prehofer, J. Wuttke, J. Andersson, H. Giese, and K. M. Göschka, "On patterns for decentralized control in self-adaptive systems," in *Software Engineering for Self-Adaptive Systems II*, pp. 76–107, Springer, 2013. (Cited on pages 8, 19, 21, 22, 24 and 46.)

[29] T. Li, Y. Liu, Y. Tian, S. Shen, and W. Mao, "A storage solution for massive iot data based on nosql," in *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pp. 50–57, IEEE, 2012. (Cited on page 9.)

[30] D. G. Páez, F. Aparicio, M. de Buenaga, and J. R. Ascanio, "Big data and iot for chronic patients monitoring," in *International Conference on Ubiquitous Computing and Ambient Intelligence*, pp. 416–423, Springer, 2014. (Cited on page 9.)

[31] L. Wang and R. Ranjan, "Processing distributed internet of things data in clouds.," *IEEE Cloud Computing*, vol. 2, no. 1, pp. 76–80, 2015. (Cited on page 9.)

[32] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015. (Cited on page 9.)

[33] C. Hochreiner, S. Schulte, S. Dustdar, and F. Lecue, "Elastic stream processing for distributed environments," *IEEE Internet Computing*, vol. 19, pp. 54–59, Nov 2015. (Cited on pages 9 and 38.)

[34] J. Soldatos, N. Kefalakis, M. Hauswirth, M. Serrano, J.-P. Calbimonte, M. Riahi, K. Aberer, P. P. Jayaraman, A. Zaslavsky, I. P. Žarko, *et al.*, "Openiot: Open source internet-of-things in the cloud," in *Interoperability and Open-Source Solutions for the Internet of Things*, pp. 13–25, Springer, 2015. (Cited on pages 9, 84 and 87.)

[35] A. Bassi, M. Bauer, M. Fiedler, T. Kramp, R. Van Kranenburg, S. Lange, and S. Meissner, "Enabling things to talk," *Designing IoT Solutions With the IoT Architectural Reference Model*, pp. 163–211, 2013. (Cited on pages 9, 84 and 87.)

[36] N. Lasierra, A. Alesanco, S. Guillén, and J. Garcia, "A three stage ontology-driven solution to provide personalized care to chronic patients at home," *Journal of biomedical informatics*, vol. 46, no. 3, pp. 516–529, 2013. (Cited on pages 9, 73, 84, 87, 113, 114 and 115.)

[37] M. Ben Alaya, S. Medjiah, T. Monteil, and K. Drira, "Toward semantic interoperability in onem2m architecture," *IEEE Communications Magazine*, vol. 53, no. 12, pp. 35–41, 2015. (Cited on pages 9, 84, 87 and 91.)

[38] J. Kim, J. Kim, D. Lee, and K.-Y. Chung, "Ontology driven interactive healthcare with wearable sensors," *Multimedia Tools and Applications*, vol. 71, no. 2, pp. 827–841, 2014. (Cited on pages 9 and 84.)

[39] M. Jarrar and R. Meersman, "Scalability and knowledge reusability in ontology modeling," *STAR*, p. 09, 2002. (Cited on page 9.)

[40] A. W. Brown, "Model driven architecture: Principles and practice," *Software and Systems Modeling*, vol. 3, no. 4, pp. 314–327, 2004. (Cited on page 16.)

[41] R. France and B. Rumpe, "Model-driven development of complex software: A research roadmap," in *2007 Future of Software Engineering*, pp. 37–54, IEEE Computer Society, 2007. (Cited on page 16.)

[42] A. Alnusair and T. Zhao, "Towards a model-driven approach for reverse engineering design patterns," in *Proceedings of the 2nd International Workshop on Transforming and Weaving Ontologies in MDE (TWOMDE 2009). Denver, Colorado, USA*, vol. 531, 2009. (Cited on page 16.)

[43] G. Hohpe and B. Woolf, *Enterprise integration patterns: Designing, building, and deploying messaging solutions*. Addison-Wesley Professional, 2004. (Cited on page 16.)

[44] W. M. van Der Aalst, A. H. Ter Hofstede, B. Kiepuszewski, and A. P. Barros, "Workflow patterns," *Distributed and parallel databases*, vol. 14, no. 1, pp. 5–51, 2003. (Cited on page 16.)

[45] E. Gamma, *Design patterns: elements of reusable object-oriented software*. Pearson Education India, 1995. (Cited on pages 16, 20, 47 and 50.)

[46] D. Atwood, "Bpm process patterns: Repeatable design for bpm process models," *BP Trends May*, 2006. (Cited on page 17.)

[47] T. Erl, *SOA design patterns*. Pearson Education, 2008. (Cited on page 17.)

[48] C. Leymann, F. Fehling, R. Retter, W. Schupeck, and P. Arbitter, *Cloud computing patterns*. Springer, 2014. (Cited on page 17.)

[49] C. Vidal, C. Fernández-Sánchez, J. Díaz, and J. Pérez, "A model-driven engineering process for autonomic sensor-actuator networks," *International Journal of Distributed Sensor Networks*, vol. 2015, p. 18, 2015. (Cited on pages 17, 25 and 27.)

[50] A. Computing, "An architectural blueprint for autonomic computing," *IBM Publication*, 2003. (Cited on page 18.)

[51] M. Parashar and S. Hariri, *Autonomic computing: concepts, infrastructure, and applications*. CRC press, 2006. (Cited on page 19.)

[52] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli, "A survey of autonomic communications," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 1, no. 2, pp. 223–259, 2006. (Cited on page 19.)

[53] P. Dazzi, F. Nidito, and M. Pasquali, "New perspectives in autonomic design patterns for stream-classification-systems," in *Proceedings of the 2007 workshop on Automating service quality: Held at the International Conference on Automated Software Engineering (ASE)*, pp. 34–37, ACM, 2007. (Cited on pages 19, 20 and 23.)

[54] A. J. Ramirez and B. H. Cheng, "Design patterns for developing dynamically adaptive systems," in *Proceedings of the 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems*, pp. 49–58, ACM, 2010. (Cited on pages 19, 20 and 23.)

[55] B. Solomon, D. Ionescu, M. Litoiu, and M. Mihaescu, "Towards a real-time reference architecture for autonomic systems," in *Proceedings of the 2007 International Workshop on Software Engineering for Adaptive and Self-Managing Systems*, p. 10, IEEE Computer Society, 2007. (Cited on pages 20, 23 and 27.)

[56] V. Mannava and T. Ramesh, "Design pattern for dynamic reconfiguration of component-based autonomic computing systems using rmi," *Procedia Technology*, vol. 6, pp. 590–597, 2012. (Cited on pages 20 and 24.)

[57] S. Frey, A. Diaconescu, and I. Demeure, "Architectural integration patterns for autonomic management systems," in *Proceedings of the 9th IEEE International Conference and Workshops on the Engineering of Autonomic and Autonomous Systems (EASe 2012). IEEE*, vol. 50, 2012. (Cited on pages 20, 22 and 25.)

[58] A. Al-Shishtawy, V. Vlassov, P. Brand, and S. Haridi, "A design methodology for self-management in distributed environments," in *Computational Science and Engineering, 2009. CSE'09. International Conference on*, vol. 1, pp. 430–436, IEEE, 2009. (Cited on pages 21, 22 and 26.)

[59] Y. Abuseta and K. Swesi, "Design patterns for self adaptive systems engineering," *arXiv preprint arXiv:1508.01330*, 2015. (Cited on pages 22 and 24.)

[60] F. A. de Oliveira Jr, R. Sharrock, and T. Ledoux, "Synchronization of multiple autonomic control loops: Application to cloud computing," in *International Conference on Coordination Languages and Models*, pp. 29–43, Springer, 2012. (Cited on pages 22, 25 and 27.)

[61] C. Krupitzer, F. M. Roth, S. VanSyckel, G. Schiele, and C. Becker, "A survey on engineering approaches for self-adaptive systems," *Pervasive and Mobile Computing*, vol. 17, pp. 184–206, 2015. (Cited on page 27.)

[62] J. Hurwitz, M. Kaufman, and A. Bowles, *Cognitive computing and big data analytics*. John Wiley & Sons, 2015. (Cited on page 28.)

[63] A. Anand and M. Singh, "Understanding knowledge management," *International Journal of Engineering Science and Technology*, vol. 3, no. 2, pp. 926–939, 2011. (Cited on pages 28 and 29.)

[64] T. H. Davenport and L. Prusak, *Working knowledge: How organizations manage what they know*. Harvard Business Press, 1998. (Cited on page 29.)

[65] J. R. Anderson, *Cognitive psychology and its implications* . WH Freeman/-Times Books/Henry Holt & Co, 1990. (Cited on page 29.)

[66] J. A. Woods and J. Cortada, *The knowledge management yearbook 2000-2001*. Routledge, 2013. (Cited on page 29.)

[67] S. M. Jasimuddin, "An integration of knowledge transfer and knowledge storage: an holistic approach," *Comput Sci Eng*, vol. 18, no. 1, pp. 37–49, 2005. (Cited on page 30.)

[68] L. M. Markus, "Toward a theory of knowledge reuse: Types of knowledge reuse situations and factors in reuse success," *Journal of management information systems*, vol. 18, no. 1, pp. 57–93, 2001. (Cited on page 30.)

[69] M. Minsky, "A framework for representing knowledge," 1975. (Cited on page 31.)

[70] M. R. Quillian, "Word concepts: A theory and simulation of some basic semantic capabilities," *Behavioral science*, vol. 12, no. 5, pp. 410–430, 1967. (Cited on page 31.)

[71] F. Van Harmelen, V. Lifschitz, and B. Porter, *Handbook of knowledge representation*, vol. 1. Elsevier, 2008. (Cited on page 31.)

[72] J. F. Sowa, "Knowledge representation: logical, philosophical, and computational foundations," 1999. (Cited on page 31.)

[73] F. Baader, *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003. (Cited on page 31.)

[74] I. Jurisica, J. Mylopoulos, and E. Yu, "Ontologies for knowledge management: an information systems perspective," *Knowledge and Information systems*, vol. 6, no. 4, pp. 380–401, 2004. (Cited on page 31.)

[75] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing?," *International journal of human-computer studies*, vol. 43, no. 5, pp. 907–928, 1995. (Cited on page 31.)

[76] I. Horrocks, "Ontologies and the semantic web," *Communications of the ACM*, vol. 51, no. 12, pp. 58–67, 2008. (Cited on page 31.)

[77] D. Fensel, "Ontologies," in *Ontologies*, pp. 11–18, Springer, 2001. (Cited on page 31.)

[78] M. Compton, P. Barnaghi, L. Bermudez, R. GarcíA-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. 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*, vol. 17, pp. 25–32, 2012. (Cited on pages 31 and 84.)

[79] M. Gagnon, "Ontology-based integration of data sources," in *Information Fusion, 2007 10th International Conference on*, pp. 1–8, IEEE, 2007. (Cited on page 31.)

[80] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *ACM Sigmod Record*, vol. 33, no. 4, pp. 65–70, 2004. (Cited on page 31.)

[81] Y. Sure, S. Staab, and R. Studer, "Ontology engineering methodology," in *Handbook on ontologies*, pp. 135–152, Springer, 2009. (Cited on page 32.)

[82] B. Stadlhofer, P. Salhofer, and A. Durlacher, "An overview of ontology engineering methodologies in the context of public administration," in *Proceedings of the 7th International Conference on Advances in Semantic Processing, IARIA, Porto, Portugal*, vol. 29, pp. 36–42, 2013. (Cited on page 32.)

[83] M. Fernández-López, "Overview of methodologies for building ontologies," 1999. (Cited on page 32.)

[84] M. Fernández-López, A. Gómez-Pérez, and N. Juristo, "Methontology: from ontological art towards ontological engineering," 1997. (Cited on page 32.)

[85] Y. Sure, S. Staab, and R. Studer, "On-to-knowledge methodology (otkm)," in *Handbook on ontologies*, pp. 117–132, Springer, 2004. (Cited on page 32.)

[86] O. Corcho, M. Fernández-López, and A. Gómez-Pérez, "Methodologies, tools and languages for building ontologies. where is their meeting point?," *Data & knowledge engineering*, vol. 46, no. 1, pp. 41–64, 2003. (Cited on page 32.)

[87] K. Kotis and G. A. Vouros, "Human-centered ontology engineering: The hcome methodology," *Knowledge and Information Systems*, vol. 10, no. 1, pp. 109–131, 2006. (Cited on page 32.)

[88] M. C. Suárez-Figueroa, A. Gomez-Perez, and M. Fernandez-Lopez, "The neon methodology for ontology engineering," in *Ontology engineering in a networked world*, pp. 9–34, Springer, 2012. (Cited on page 32.)

[89] M. E. Warkentin, L. Sayeed, and R. Hightower, "Virtual teams versus face-to-face teams: an exploratory study of a web-based conference system," *Decision Sciences*, vol. 28, no. 4, pp. 975–996, 1997. (Cited on page 33.)

[90] K. Yaakub, R. Shaari, S. A. Panatik, and A. Rahman, "Towards an understanding of the effect of core self-evaluations and knowledge sharing behaviour," *International Journal of Applied Psychology*, vol. 3, no. 1, pp. 13–18, 2013. (Cited on page 33.)

[91] T. Berners-Lee, J. Hendler, O. Lassila, *et al.*, "The semantic web," *Scientific american*, vol. 284, no. 5, pp. 28–37, 2001. (Cited on pages 33 and 114.)

[92] J. Hendler and F. van Harmelen, "The semantic web: webizing knowledge representation," *Foundations of Artificial Intelligence*, vol. 3, pp. 821–839, 2008. (Cited on page 33.)

[93] C. Bizer and A. Schultz, "The berlin sparql benchmark," 2009. (Cited on page 33.)

[94] S. Schaffert, F. Bry, J. Baumeister, and M. Kiesel, "Semantic wikis," *IEEE software*, vol. 25, no. 4, pp. 8–11, 2008. (Cited on page 33.)

[95] Z. A. Bhatti, S. Baile, and H. M. Yasin, "The success of corporate wiki systems: an end user perspective," in *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*, pp. 134–143, ACM, 2011. (Cited on page 33.)

[96] B. Hoenderboom and P. Liang, "A survey of semantic wikis for requirements engineering," *SEARCH, University of Groningen, Tech. Rep*, 2009. (Cited on page 33.)

[97] F. Bry, S. Schaffert, D. Vrandečić, and K. Weiand, "Semantic wikis: Approaches, applications, and perspectives," in *Reasoning Web International Summer School*, pp. 329–369, Springer, 2012. (Cited on page 33.)

[98] M. Krötzsch, D. Vrandečić, and M. Völkel, "Semantic mediawiki," in *International semantic web conference*, pp. 935–942, Springer, 2006. (Cited on pages 33 and 93.)

[99] M. Buffa, F. Gandon, G. Ereteo, P. Sander, and C. Faron, "Sweetwiki: A semantic wiki," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 1, pp. 84–97, 2008. (Cited on page 33.)

[100] P. NBD, "Nist big data interoperability framework," 2015. (Cited on page 34.)

[101] Y.-S. Kang, I.-H. Park, J. Rhee, and Y.-H. Lee, "Mongodb-based repository design for iot-generated rfid/sensor big data," *IEEE Sensors Journal*, vol. 16, no. 2, pp. 485–497, 2016. (Cited on page 34.)

[102] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, "Trends in big data analytics," *Journal of Parallel and Distributed Computing*, vol. 74, no. 7, pp. 2561–2573, 2014. (Cited on page 34.)

[103] K. Grolinger, W. A. Higashino, A. Tiwari, and M. A. Capretz, "Data management in cloud environments: Nosql and newsql data stores," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 2, no. 1, p. 1, 2013. (Cited on page 34.)

[104] R. Hecht and S. Jablonski, "Nosql evaluation," in *International conference on cloud and service computing*, pp. 336–41, IEEE, 2011. (Cited on page 34.)

[105] C. Cecchinel, M. Jimenez, S. Mosser, and M. Riveill, "An architecture to support the collection of big data in the internet of things," in *2014 IEEE World Congress on Services*, pp. 442–449, IEEE, 2014. (Cited on page 34.)

[106] L. Jiang, L. Da Xu, H. Cai, Z. Jiang, F. Bu, and B. Xu, "An iot-oriented data storage framework in cloud computing platform," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1443–1451, 2014. (Cited on pages 34 and 38.)

[107] M. Chen, Y. Zhang, Y. Li, M. M. Hassan, and A. Alamri, "Aiwac: affective interaction through wearable computing and cloud technology," *IEEE Wireless Communications*, vol. 22, no. 1, pp. 20–27, 2015. (Cited on page 35.)

[108] P. A. Prakashbhai and H. M. Pandey, "Inference patterns from big data using aggregation, filtering and tagging-a survey," in *Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference-*, pp. 66–71, IEEE, 2014. (Cited on page 35.)

[109] A. McAfee, E. Brynjolfsson, T. H. Davenport, D. Patil, and D. Barton, "Big data," *The management revolution. Harvard Bus Rev*, vol. 90, no. 10, pp. 61–67, 2012. (Cited on page 35.)

[110] H. Chen, S. Compton, and O. Hsiao, "Diabeticlink: a health big data system for patient empowerment and personalized healthcare," in *International Conference on Smart Health*, pp. 71–83, Springer, 2013. (Cited on page 35.)

[111] A. O'Driscoll, J. Daugelaite, and R. D. Sleator, "'big data', hadoop and cloud computing in genomics," *Journal of biomedical informatics*, vol. 46, no. 5, pp. 774–781, 2013. (Cited on page 35.)

[112] M. Chen, S. Mao, Y. Zhang, and V. C. Leung, *Big data: related technologies, challenges and future prospects*. Springer, 2014. (Cited on page 35.)

[113] I. Anagnostopoulos, S. Zeadally, and E. Exposito, "Handling big data: research challenges and future directions," *The Journal of Supercomputing*, vol. 72, no. 4, pp. 1494–1516, 2016. (Cited on page 36.)

[114] M. Barlow, *Real-Time Big Data Analytics: Emerging Architecture*. " O'Reilly Media, Inc.", 2013. (Cited on page 36.)

[115] M. Almorsy, J. Grundy, and A. S. Ibrahim, "Collaboration-based cloud computing security management framework," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pp. 364–371, IEEE, 2011. (Cited on page 38.)

[116] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010. (Cited on page 38.)

[117] P. Mell and T. Grance, "The nist definition of cloud computing," 2011. (Cited on page 38.)

[118] F. Liu, J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger, and D. Leaf, "Nist cloud computing reference architecture," *NIST special publication*, vol. 500, no. 2011, p. 292, 2011. (Cited on pages 38 and 88.)

[119] J. Zhou, T. Leppänen, E. Harjula, M. Ylianttila, T. Ojala, C. Yu, and H. Jin, "Cloudthings: A common architecture for integrating the internet of things with cloud computing," in *Computer Supported Cooperative Work in Design (CSCWD), 2013 IEEE 17th International Conference on*, pp. 651–657, IEEE, 2013. (Cited on page 38.)

[120] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016. (Cited on page 38.)

[121] F. Tao, Y. Cheng, L. Da Xu, L. Zhang, and B. H. Li, "Cciot-cmfg: cloud computing and internet of things-based cloud manufacturing service system," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1435–1442, 2014. (Cited on page 38.)

[122] S. Kyriazakos, B. Anggorojati, N. Prasad, C. Vallati, E. Mingozzi, G. Tanganelli, N. Buonaccorsi, N. Valdambrini, N. Zonidis, G. Labropoulous, *et al.*, "Betaas platform–a things as a service environment for future m2m marketplaces," in *Internet of Things. User-Centric IoT*, pp. 305–313, Springer, 2015. (Cited on pages 38 and 85.)

[123] Z. Zheng, J. Zhu, and M. R. Lyu, "Service-generated big data and big data-as-a-service: an overview," in *2013 IEEE international congress on Big Data*, pp. 403–410, IEEE, 2013. (Cited on page 38.)

[124] S. K. Sowe, T. Kimata, M. Dong, and K. Zettsu, "Managing heterogeneous sensor data on a big data platform: Iot services for data-intensive science," in *Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International*, pp. 295–300, IEEE, 2014. (Cited on pages 38, 85 and 87.)

[125] M. Ribeiro, K. Grolinger, and M. A. Capretz, "Mlaas: Machine learning as a service," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 896–902, Dec 2015. (Cited on page 38.)

[126] S. Xu and W. Zhang, "Knowledge as a service and knowledge breaching," in *2005 IEEE International Conference on Services Computing (SCC'05) Vol-1*, vol. 1, pp. 87–94, IEEE, 2005. (Cited on page 39.)

[127] R. Abdullah, Z. D. Eri, and A. M. Talib, "A model of knowledge management system for facilitating knowledge as a service (kaas) in cloud computing environment," in *2011 International Conference on Research and Innovation in Information Systems*, pp. 1–4, IEEE, 2011. (Cited on pages 39 and 88.)

[128] Y. Qirui, "Kaas-based intelligent service model in agricultural expert system," in *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, pp. 2678–2680, IEEE, 2012. (Cited on pages 39, 40 and 88.)

[129] S. Kannimuthu, K. Premalatha, and S. Shankar, "Investigation of high utility itemset mining in service oriented computing: Deployment of knowledge as a service in e-commerce," in *2012 Fourth International Conference on*

Advanced Computing (ICoAC), pp. 1–8, IEEE, 2012. (Cited on pages 39, 40 and 88.)

[130] N. C. Q. Lino, C. d. A. Siebra, M. Amaro, and A. Tate, "Emergencygrid–planning in convergence environments," *SPARK 2012*, p. 56, 2012. (Cited on pages 39, 40 and 88.)

[131] S. Maru, G. G. Koch, M. Stender, D. Clark, L. Gibowski, H. Petri, A. D. White, and R. J. Simpson, "Antidiabetic drugs and heart failure risk in patients with type 2 diabetes in the uk primary care setting," *Diabetes care*, vol. 28, no. 1, pp. 20–26, 2005. (Cited on page 42.)

[132] N. I. on Drug Abuse and U. S. of America, "Comorbidity: Addiction and other mental illnesses," 2008. (Cited on page 44.)

[133] P. Lalanda, "Two complementary patterns to build multi-expert systems," in *Pattern Languages of Programs*, 1997. (Cited on page 46.)

[134] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys (CSUR)*, vol. 35, no. 2, pp. 114–131, 2003. (Cited on page 47.)

[135] R. C. Atkinson and R. M. Shiffrin, "Human memory: A proposed system and its control processes," *Psychology of learning and motivation*, vol. 2, pp. 89–195, 1968. (Cited on page 53.)

[136] E. Tulving, "How many memory systems are there?," *American psychologist*, vol. 40, no. 4, p. 385, 1985. (Cited on page 53.)

[137] J. R. Manns, R. O. Hopkins, and L. R. Squire, "Semantic memory and the human hippocampus," *Neuron*, vol. 38, no. 1, pp. 127–133, 2003. (Cited on page 53.)

[138] A. Soliman, V. DeSanctis, M. Yassin, R. Elalaily, and N. E. Eldarsy, "Continuous glucose monitoring system and new era of early diagnosis of diabetes in high risk groups," *Indian journal of endocrinology and metabolism*, vol. 18, no. 3, p. 274, 2014. (Cited on page 56.)

[139] Y. Liu, G. Tong, W. Tong, L. Lu, and X. Qin, "Can body mass index, waist circumference, waist-hip ratio and waist-height ratio predict the presence of multiple metabolic risk factors in chinese subjects?," *BMC Public Health*, vol. 11, no. 1, p. 1, 2011. (Cited on page 59.)

[140] M. Compton, C. Henson, L. Lefort, H. Neuhaus, and A. Sheth, "A survey of the semantic specification of sensors," in *Proceedings of the 2nd International Conference on Semantic Sensor Networks-Volume 522*, pp. 17–32, CEUR-WS. org, 2009. (Cited on page 73.)

[141] B. Farran, A. M. Channanath, K. Behbehani, and T. A. Thanaraj, "Predictive models to assess risk of type 2 diabetes, hypertension and comorbidity: machine-learning algorithms and validation using national health data from kuwait?a cohort study," *BMJ open*, vol. 3, no. 5, p. e002457, 2013. (Cited on pages 77 and 92.)

[142] E. Chiauzzi, C. Rodarte, and P. DasMahapatra, "Patient-centered activity monitoring in the self-management of chronic health conditions," *BMC medicine*, vol. 13, no. 1, p. 1, 2015. (Cited on page 82.)

[143] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, 2010. (Cited on page 82.)

[144] S. Patel, H. Park, P. Bonato, L. Chan, and M. Rodgers, "A review of wearable sensors and systems with application in rehabilitation," *Journal of neuroengineering and rehabilitation*, vol. 9, no. 1, p. 1, 2012. (Cited on page 82.)

[145] S. C. Mukhopadhyay, "Wearable sensors for human activity monitoring: A review," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1321–1330, 2015. (Cited on page 82.)

[146] J. Penders, M. Altini, C. Van Hoof, and E. Dy, "Wearable sensors for healthier pregnancies," *Proceedings of the IEEE*, vol. 103, no. 2, pp. 179–191, 2015. (Cited on page 82.)

[147] M. A. Nicolelis, "Brain–machine interfaces to restore motor function and probe neural circuits," *Nature Reviews Neuroscience*, vol. 4, no. 5, pp. 417–422, 2003. (Cited on page 83.)

[148] K. Pretz, "Better Health Care Through Data: How health analytics could contain costs and improve care." `http://theinstitute.ieee.org/ns/quarterly_issues/tisep14.pdf`, 2014. [Online; accessed 15-September-2016]. (Cited on page 83.)

[149] W. Kim, S. Lim, J. Ahn, J. Nah, and N. Kim, "Integration of ieee 1451 and hl7 exchanging information for patients? sensor data," *Journal of medical systems*, vol. 34, no. 6, pp. 1033–1041, 2010. (Cited on pages 84 and 87.)

[150] Á. Ruiz-Zafra, M. Noguera, and K. Benghazi, "Towards a model-driven approach for sensor management in wireless body area networks," in *International Conference on Internet and Distributed Computing Systems*, pp. 335–347, Springer, 2014. (Cited on page 84.)

[151] N. Bui, "Internet of things architecture (iot-a), project deliverable d1. 1-sota report on existing integration frameworks/architectures for wsn, rfid and other emerging iot related technology," 2014. (Cited on page 84.)

[152] T. J. NEC, S. Meissner, G. Völksen, C. Kleegrewe, and S. Becher, "Internet-of-things architecture iot-a project deliverable d2. 6," (Cited on page 84.)

[153] X. Xu, S. Huang, Y. Chen, K. Browny, I. Halilovicy, and W. Lu, "Tsaaas: Time series analytics as a service on iot," in *Web Services (ICWS), 2014 IEEE International Conference On*, pp. 249–256, IEEE, 2014. (Cited on pages 85 and 87.)

[154] E. Mingozzi, G. Tanganelli, C. Vallati, and V. Di Gregorio, "An open framework for accessing things as a service," in *Wireless Personal Multimedia Communications (WPMC), 2013 16th International Symposium on*, pp. 1–5, IEEE, 2013. (Cited on pages 85 and 87.)

[155] I. Mendia, "Semantics in betaas," 2014. (Cited on page 85.)

[156] A. Forkan, I. Khalil, and Z. Tari, "Cocamaal: A cloud-oriented context-aware middleware in ambient assisted living," *Future Generation Computer Systems*, vol. 35, pp. 114–127, 2014. (Cited on pages 85, 86 and 87.)

[157] P. Jiang, J. Winkley, C. Zhao, R. Munnoch, G. Min, and L. T. Yang, "An intelligent information forwarder for healthcare big data systems with distributed wearable sensors," *IEEE Systems Journal*, vol. 10, pp. 1147–1159, Sept 2016. (Cited on pages 86 and 87.)

[158] K. Grolinger, E. Mezghani, M. A. Capretz, and E. Exposito, "Collaborative knowledge as a service applied to the disaster management domain," *International Journal of Cloud Computing 2*, vol. 4, no. 1, pp. 5–27, 2015. (Cited on page 88.)

[159] R. A. S. NIST Big Data Public Working Group, "Nist big data interoperability framework: Volume 6, reference architecture," 2015. (Cited on page 88.)

[160] E. Mezghani, E. Exposito, K. Drira, M. Da Silveira, and C. Pruski, "A semantic big data platform for integrating heterogeneous wearable data in healthcare," *Journal of medical systems*, vol. 39, no. 12, pp. 1–8, 2015. (Cited on page 90.)

[161] M. Lichman, "Uci machine learning repository http://archive. ics. uci. edu/ml. irvine, ca: University of california, school of information and computer science, 2013," (Cited on page 97.)

[162] A. Wright and D. F. Sittig, "A four-phase model of the evolution of clinical decision support architectures," *International journal of medical informatics*, vol. 77, no. 10, pp. 641–649, 2008. (Cited on page 111.)

[163] E. S. Berner and T. J. La Lande, "Overview of clinical decision support systems," in *Clinical decision support systems*, pp. 3–22, Springer, 2007. (Cited on page 111.)

[164] P. A. De Clercq, J. A. Blom, H. H. Korsten, and A. Hasman, "Approaches for creating computer-interpretable guidelines that facilitate decision support," *Artificial intelligence in medicine*, vol. 31, no. 1, pp. 1–27, 2004. (Cited on page 111.)

[165] M. A. Grando, D. Glasspool, and A. Boxwala, "Argumentation logic for the flexible enactment of goal-based medical guidelines," *Journal of biomedical informatics*, vol. 45, no. 5, pp. 938–949, 2012. (Cited on pages 111, 112, 113 and 115.)

[166] Z. Huang, X. Lu, and H. Duan, "Using recommendation to support adaptive clinical pathways," *Journal of Medical Systems*, vol. 36, no. 3, pp. 1849–1860, 2012. (Cited on pages 111, 112, 113 and 115.)

[167] A. González-Ferrer, A. Ten Teije, J. Fdez-Olivares, and K. Milian, "Automated generation of patient-tailored electronic care pathways by translating computer-interpretable guidelines into hierarchical task networks," *Artificial intelligence in medicine*, vol. 57, no. 2, pp. 91–109, 2013. (Cited on pages 111, 112, 113 and 115.)

[168] G. Milla-Millán, J. Fdez-Olivares, I. Sánchez-Garzón, D. Prior, and L. Castillo, "Knowledge-driven adaptive execution of care pathways based on continuous planning techniques," in *Process Support and Knowledge Representation in Health Care*, pp. 42–55, Springer, 2013. (Cited on pages 112, 113 and 115.)

[169] I. Sánchez-Garzón, G. Milla-Millán, and J. Fernández-Olivares, "Context-aware generation and adaptive execution of daily living care pathways," in *International Workshop on Ambient Assisted Living*, pp. 362–370, Springer, 2012. (Cited on pages 112 and 115.)

[170] D. Riaño, F. Real, J. A. López-Vallverdú, F. Campana, S. Ercolani, P. Mecocci, R. Annicchiarico, and C. Caltagirone, "An ontology-based personalization of health-care knowledge to support clinical decisions for chronically ill patients," *Journal of biomedical informatics*, vol. 45, no. 3, pp. 429–446, 2012. (Cited on pages 112, 113 and 115.)

[171] A. Emerencia, L. Van Der Krieke, S. Sytema, N. Petkov, and M. Aiello, "Generating personalized advice for schizophrenia patients," *Artificial intelligence in medicine*, vol. 58, no. 1, pp. 23–36, 2013. (Cited on pages 112, 113 and 115.)

[172] D. A. Alexandrou, I. E. Skitsas, and G. N. Mentzas, "A holistic environment for the design and execution of self-adaptive clinical pathways," *IEEE Transactions on Information Technology in Biomedicine*, vol. 15, no. 1, pp. 108–118, 2011. (Cited on pages 113, 114 and 115.)

[173] W. Yao and A. Kumar, "Conflexflow: Integrating flexible clinical pathways into clinical decision support systems using context and rules," *Decision Support Systems*, vol. 55, no. 2, pp. 499–515, 2013. (Cited on pages 113, 114 and 115.)

[174] T. Heath and C. Bizer, "Linked data: Evolving the web into a global data space," *Synthesis lectures on the semantic web: theory and technology*, vol. 1, no. 1, pp. 1–136, 2011. (Cited on page 114.)

[175] A. Jentzsch, O. Hassanzadeh, C. Bizer, B. Andersson, and S. Stephens, "Enabling tailored therapeutics with linked data," in *Proceedings of the 2nd Workshop on Linked Data on the Web (LDOW2009)*, pp. 1–6, 2009. (Cited on pages 114 and 115.)

[176] M. Samwald, A. Jentzsch, C. Bouton, C. S. Kallesøe, E. Willighagen, J. Hajagos, M. S. Marshall, E. Prud'hommeaux, O. Hassanzadeh, E. Pichler, *et al.*, "Linked open drug data for pharmaceutical research and development," *Journal of cheminformatics*, vol. 3, no. 1, p. 19, 2011. (Cited on page 114.)

[177] S. Ayvaz, J. Horn, O. Hassanzadeh, Q. Zhu, J. Stan, N. P. Tatonetti, S. Vilar, M. Brochhausen, M. Samwald, M. Rastegar-Mojarad, *et al.*, "Toward a complete dataset of drug–drug interaction information from publicly available sources," *Journal of biomedical informatics*, vol. 55, pp. 206–217, 2015. (Cited on page 114.)

[178] A. Callahan, J. Cruz-Toledo, and M. Dumontier, "Ontology-based querying with bio2rdf's linked open data," *Journal of biomedical semantics*, vol. 4, no. Suppl 1, p. S1, 2013. (Cited on page 114.)

[179] A. Ostankov, F. Röhrbein, and U. Waltinger, "Linkedhealthanswers: Towards linked data-driven question answering for the health care domain," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)* (N. Calzolari, K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, eds.), (Reykjavik, Iceland), pp. 2613–2620, European Language Resources Association (ELRA), May 2014. ACL Anthology Identifier: L14-1691. (Cited on page 115.)

[180] A. Khalili and B. Sedaghati, "Semantic medical prescriptions–towards intelligent and interoperable medical prescriptions," in *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pp. 347–354, IEEE, 2013. (Cited on page 115.)

[181] P. J. O'Connor, J. M. Sperl-Hillen, P. E. Johnson, and W. A. Rush, "Identification, classification, and frequency of medical errors in outpatient diabetes care," 2005. (Cited on page 116.)

[182] N. Lasierra Beamonte, *An ontology-driven architecture for data integration and management in home-based telemonitoring scenarios*. PhD thesis, Universidad de Zaragoza, 2012. (Cited on page 116.)

[183] S. Wang and R. A. Noe, "Knowledge sharing: A review and directions for future research," *Human Resource Management Review*, vol. 20, no. 2, pp. 115–131, 2010. (Cited on page 118.)

[184] R. J. Brachman and H. J. Levesque, "The tractability of subsumption in frame-based description languages," in *AAAI*, vol. 84, pp. 34–37, 1984. (Cited on page 120.)

[185] E. Mezghani, E. Exposito, and K. Drira, "A collaborative methodology for tacit knowledge management: Application to scientific research," *Future Generation Computer Systems*, vol. 54, pp. 450–455, 2016. (Cited on page 120.)

[186] M. Ghallab, D. Nau, and P. Traverso, *Automated planning: theory & practice*. Elsevier, 2004. (Cited on page 120.)

[187] E. Mezghani, M. Da Silveira, C. Pruski, E. Exposito, and K. Drira, "An ontology-driven adaptive system for the patient treatment management,"

in *Twenty-Eighth International Conference on Software Engineering and Knowledge Engineering*, pp. 329–332, Knowledge Systems Institute, 2016. (Cited on page 120.)

[188] S. E. Inzucchi, R. M. Bergenstal, J. B. Buse, M. Diamant, E. Ferrannini, M. Nauck, A. L. Peters, A. Tsapas, R. Wender, and D. R. Matthews, "Management of hyperglycemia in type 2 diabetes: a patient-centered approach position statement of the american diabetes association (ada) and the european association for the study of diabetes (easd)," *Diabetes care*, vol. 35, no. 6, pp. 1364–1379, 2012. (Cited on pages 121, 124, 126 and 129.)

[189] A. Baker, "Crossing the quality chasm: A new health system for the 21st century," *BMJ*, vol. 323, no. 7322, p. 1192, 2001. (Cited on page 126.)

[190] R. E. Say and R. Thomson, "The importance of patient preferences in treatment decisions–challenges for doctors," *Bmj*, vol. 327, no. 7414, pp. 542–545, 2003. (Cited on page 127.)

[191] S. W. Lahiri, "Management of type 2 diabetes: what is the next step after metformin?," *Clinical Diabetes*, vol. 30, no. 2, pp. 72–75, 2012. (Cited on pages 131 and 132.)