Universidad Politécnica de Madrid

Escuela Técnica Superior de
Ingeniería y Sistemas de
Telecomunicación

Toward Context Awareness in the Cooperation of Underwater Robots

DOCTORAL THESIS

Xin Li

Master in Systems and Services Engineering

for the Information Society

Supervisor:

Prof. PhD. José-Fernán Martínez Ortega

2017

**DOCTORADO EN INGENIERÍA DE SISTEMAS Y SERVICIOS
PARA LA SOCIEDAD DE LA INFORMACIÓN**

Tesis Doctoral		
Título	Toward context awareness in the cooperation of underwater robots	
Autor	Xin Li	
Co-Director	José-Fernán Martínez Ortega	VºBº.
Co-Directora		
Tribunal		
Presidente	Tomás Robles Valladares	
Secretario	Lourdes López Santidrián	
Vocal	Mikael Ekström	
Vocal	Carlos García Rubio	
Vocal	Miguel Ángel López Carmona	
Suplente	Kristoffer Nyborg Gregertsen	
Suplente	Rodolfo Haber Guerra	
Lugar y fecha de lectura	E.T.S.I. y Sistemas de Telecomunicación (U.P.M.)	
Calificación		

El Presidente

El secretario

Los vocales

Tesis Doctoral para la obtención del título de Doctor
por la Universidad Politécnica de Madrid

2017

Acknowledgements

This thesis would not have been accomplished without the help, care, and support of many people. The four-year PhD study full with joys and sorrows is one of the best journeys up to now in my life. By taking the chance, here, I would like to express my sincerest gratitude and thanks to all the lovely people I met along this journey.

First and foremost, I would like to thank my supervisor Prof. José-Fernán Martínez-Ortega, who provided me the precious opportunity to pursue my PhD study under his supervision in the group of Next Generation Networks and Services (GRyS) at UPM. His knowledge, wisdom, rigor, and extraordinary supervision have inspired me a lot. Without his invaluable encouragement and guidance, it would not be possible for me to turn to be a professional researcher from a fresh graduate student.

Special thanks are also paid to my brilliant professors and colleagues in the group. I will cherish all the enjoyable moments and fruitful discussions.

Last but not the least, I want to thank my family for their unconditional love and support to me. They are the ones that I know I can share everything, laughs or tears, with. Especially, in the hardest time of the PhD study, they consoled my soul and accompanied me through the tough days. Their positive attitude and sense of humor motivated me a lot. They always encouraged me in those difficult times using the saying, *When Life Gives You Lemons, Make Lemonade*. My husband deserves my deepest thanks for his love, understanding, support, and patience. Thanks for not only being my partner but also being my best friend.

This thesis is dedicated to the unforgettable journey in Madrid, from 2014.11 to 2017.11.

Abstract

Nowadays, underwater robots are becoming more and more popular to carry out maritime operations, such as oil spill detection and berm construction. Knowing and understanding the physical environment where specific operations are to take place is key for accomplishing effectively and efficiently such operations and associated activities. On the other hand, when the considered activities become more complex demanding a dynamic setup, the knowledge of environment can allow enhancing performance and make adaptations. Nonetheless, it is very difficult for robots or operators to know the underwater environment, which is highly dynamic and uncertain, a priori, or even in a rough way. Facilitating context awareness, which is a capability for entities to be context-aware, in the cooperation of underwater robots still remains a challenge.

The aim of this thesis is to present a comprehensive study on delivering context awareness in the cooperation of underwater robots and overcome the most challenging problems of this topic. To achieve this objective, this thesis is carried out centering on three main research areas.

The first area aims to provide a general solution for delivering context awareness in underwater robots. An architectural proposal of a context-aware framework for underwater vehicles is presented in this thesis. The proposed context-aware framework provides a complete and well-defined context management, including context acquisition, context modeling, context reasoning, context distribution, and context dissemination. It can be an enabler of context awareness to be integrated into existing underwater robotic middleware architectures. Services provided by this framework can be exploited in different ways, such as being used by robots to understand the surrounding and for operators to conceive mission plans.

The second area concentrates on properly modeling context information that is necessarily exchanged between robots. In this area, three main contributions are made. Firstly, a fuzzy ontology development methodology (FODM) is proposed for guiding the building of fuzzy ontologies from scratch. Secondly, an ontology proposal, named the SWARMs ontology, is presented and implemented. The SWARMs ontology consists of a core ontology and four domain-specific ontologies, including mission & planning, robotic vehicle, environment recognition & sensing, and communication & networking. It is also able to be extended with fuzzy and probabilistic annotations to represent context uncertainty. Especially, the Probabilistic Web Ontology Language (PR-OWL) ontology is adopted to express context uncertainty based on the Multi-Entity Bayesian Network (MEBN) theory. In this way, the SWARMs ontology can not only

present a comprehensive and principled representation of context and its associated uncertainty but also provide support for uncertainty reasoning. Finally, a proposal of applying the Stochastic Reduced Order Model (SROM) algorithm to quantify uncertainties propagated in mathematic relationships in the SWARMs ontology is presented. This proposal can guarantee a considerable degree of accuracy in approximating the statistics of uncertain ontological elements but with much fewer calculations. It is worth noting that this proposal is general enough to be applied to quantify uncertainties in any mathematics-embedded ontologies.

The last area under investigation focuses on how to effectively reason about context information and its uncertainty in the underwater robot field. None of the existing context reasoning methods can individually meet the reasoning requirements in the underwater robot field. Therefore, a hybrid context reasoning mechanism is proposed in this thesis. The proposal is to loosely couple three different context reasoning methods, namely, the ontological, rule-based, and MEBN reasoning techniques. With the combination of the different reasoning methods, it is flexible to mitigate each reasoning's weaknesses by using others' strengths. A set of Java Application Programming Interfaces (APIs) is implemented to realize the hybrid context reasoning proposal. The implementation provides simple interfaces to use the existing context reasoners, including the Jena OWL reasoner, Pellet reasoner, and the UnBBayes-MEBN reasoner, to provide reasoning capabilities. In addition, the implementation of this proposal is validated in terms of usefulness. A preliminary performance analysis on the hybrid context reasoning mechanism is also provided and it shows that the hybrid context reasoner can provide reasoning capabilities within an acceptable time span.

Resumen

Hoy en día los robots submarinos están creciendo en popularidad en la realización de operaciones marítimas tales como la detección de derrames de petróleo y la preparación del fondo marino para la construcción de diferentes infraestructuras. La clave para poder realizar con eficacia y eficiencia estas operaciones y sus actividades asociadas pasa por comprender y conocer el entorno físico en el que se realizan. Por otra parte el conocimiento del entorno puede permitir mejorar el rendimiento y realizar adaptaciones cuando dichas actividades se vuelvan más complejas y requieran de una configuración dinámica. Sin embargo es muy difícil que los robots y los operadores conozcan el entorno submarino, aunque sea de forma aproximada, por ser este un entorno a priori dinámico e incierto. Así pues facilitar el conocimiento del contexto de las operaciones, logrando que las entidades que participan en ellas sean conscientes del mismo, sigue siendo un desafío para la cooperación de las actividades de los robots submarinos.

El objetivo de esta tesis es presentar un estudio exhaustivo sobre la transferencia de conocimiento del contexto para la cooperación de robots submarinos y superar los problemas más desafiantes sobre este tema. Para alcanzar este objetivo la tesis se ha centrado en tres áreas principales de investigación.

La primera de estas áreas tiene como objetivo proporcionar una solución general para dotar de consciencia del contexto a los robots submarinos. Así pues se presenta en esta tesis una propuesta de arquitectura de un marco de consciencia del contexto para vehículos submarinos. El marco propuesto proporciona una gestión completa y bien definida del contexto, incluyendo su adquisición, modelado, razonamiento, distribución y difusión. Este marco puede ser el facilitador de la consciencia del contexto integrada en las arquitecturas de intermediación que ya existen en el ámbito de la robótica submarina. Los servicios que ofrece este marco pueden explotarse de diferentes formas, ya sea por los robots para comprender el entorno o por los operadores para concebir los planes de la misión.

La segunda área se concentra en el modelado adecuado de la información de contexto que los robots necesitan intercambiar entre ellos. En esta área se realizan tres contribuciones principales. En primer lugar se propone una metodología de desarrollo de ontologías difusas (FODM, *Fuzzy Ontology Development Methodology*) que sirva de guía para la creación de ontologías difusas desde cero. En segundo lugar se presenta e implementa una propuesta de ontología, la ontología SWARMS. Consiste en una ontología principal y cuatro ontologías de

dominio específico correspondientes a la misión y planificación, a los vehículos robóticos, al reconocimiento y detección del entorno, y a las redes y comunicaciones. Además puede ser extendida con anotaciones difusas y probabilísticas representando así la incertidumbre del contexto. En particular se ha adoptado PR-OWL para expresar la incertidumbre del contexto basándose en la teoría de redes bayesianas de entidades múltiples (MEBN, *Multi-Entity Bayesian Networks*). Así la ontología SWARMS no está limitada a la representación exhaustiva y principal del contexto con su incertidumbre asociada, sino que además puede dar soporte al razonamiento con incertidumbre. Finalmente se presenta una propuesta para aplicar el algoritmo del modelo estocástico de orden reducido (SROM, *Stochastic Reduced Order Model*), siendo usado para cuantificar las incertidumbres propagadas en las relaciones matemáticas de la ontología SWARMS. Esta propuesta puede garantizar un grado de precisión considerable en la aproximación de las estadísticas de los elementos ontológicos con incertidumbre, usando en el proceso muchos menos cálculos que de forma directa. Vale la pena destacar que esta propuesta es lo suficientemente genérica como para ser aplicada en la cuantificación de incertidumbres en cualquier ontología que tenga matemáticas incrustadas.

La tercera área de investigación se centra en cómo razonar de forma efectiva sobre la información de contexto y su incertidumbre aplicándose al campo de la robótica submarina. Ninguno de los métodos existentes de razonamiento de contexto puede en la actualidad cubrir individualmente los requisitos de la robótica submarina. Por tanto en esta tesis se propone un mecanismo híbrido de razonamiento de contexto. Esta propuesta se basa en un acoplamiento ligero entre tres métodos diferentes de razonamiento de contexto, que en concreto son los basados en técnicas ontológicas, los basados en reglas, y los que emplean MEBN. Empleando una combinación de los diferentes métodos de razonamiento es posible mitigar las debilidades de cada razonador recurriendo a las puntos fuertes de los demás. Se ha implementado un conjunto de APIs Java para realizar la propuesta del razonador híbrido de contexto. La implementación proporciona interfaces sencillas para utilizar los razonadores de contexto existentes, tales como el razonador OWL de Jena, el razonador Pellet, y el razonador UnBBayes-MEBN. Además la implementación de esta propuesta se ha validado en términos de utilidad. Se ha incluido un análisis preliminar del rendimiento del mecanismo híbrido de razonamiento de contexto, mostrando que sus capacidades de razonamiento se pueden proporcionar en un espacio de tiempo aceptable.

Table of content

Acknowledgements.....	I
Abstract	II
Resumen	IV
Table of content.....	VI
List of figures	IX
List of tables	XIII
List of equations	XIV
1. Introduction.....	1
1.1. Context awareness	2
1.2. The importance of context awareness in the cooperation of underwater robots	4
1.3. Research framework.....	4
1.4. Problem statement and thesis objectives	6
1.5. Research contributions and thesis structure	8
2. Background and related work	10
2.1. Context modeling.....	11
2.1.1. Existing context modeling methods.....	11
2.1.2. Uncertainty modeling	16
2.1.3. Context modeling in the underwater robot field	22
2.1.4. Summary	23
2.2. Context reasoning.....	24
2.2.1. Review of existing context reasoning methods	24
2.2.2. Summary	25
2.3. Context-aware framework in the underwater field	26
2.3.1. Existing context-aware frameworks.....	26
2.3.2. Summary	40
3. The context-aware framework for the cooperation of underwater robots	43
4. Context modeling for the cooperation of underwater robots	48
4.1. Ontology development methodologies	49
4.1.1. Existing ontology development methodologies	49
4.1.2. A fuzzy ontology development methodology (FODM)	51

4.1.3.	Uncertainty Modeling Process for Semantic Technology (UMP-ST).....	60
4.2.	The SWARMs ontology.....	62
4.2.1.	The domain and scope of ontology	62
4.2.2.	Design requirements.....	63
4.2.3.	Description of the SWARMs ontology	65
4.2.4.	Formalization of the SWARMs ontology.....	73
4.2.5.	Fuzzy extensions: an example of seabed characterization	75
4.2.6.	Probabilistic extensions: an example of oil spill monitoring	79
4.3.	Uncertainty quantification in mathematics-embedded ontologies using Stochastic Reduced Order Model	90
4.3.1.	Mathematic models in ontologies	90
4.3.2.	Stochastic Reduced Order Model (SROM)	91
4.3.3.	Application of SROM in mathematics-embedded ontologies.....	94
4.3.4.	A case study	95
5.	Context reasoning for the cooperation of underwater robots	103
5.1.	The hybrid context reasoning mechanism.....	104
5.1.1.	Ontological reasoning	105
5.1.2.	Rule-based reasoning.....	106
5.1.3.	MEBN reasoning.....	107
5.2.	A preliminary proof of concept	108
5.2.1.	Ontological reasoning results.....	109
5.2.2.	Rule-based reasoning results	111
5.2.3.	MEBN reasoning results	113
6.	Implementation and validation	118
6.1.	Implementation scope and purpose	119
6.2.	The implementation of the proposed SWARMs ontology	120
6.3.	The implementation of the context reasoner component.....	124
6.3.1.	Specifications for the use of different context reasoning	125
6.3.2.	Implementation of different context reasoning capabilities in the context reasoner.....	128
6.4.	Validation of the hybrid context reasoning mechanism.....	131
6.4.1.	Test on ontological reasoning	131
6.4.2.	Test on rule-based reasoning	133
6.4.3.	Test on MEBN reasoning.....	136
7.	Conclusions and future work.....	140

POLITÉCNICA

7.1.	Conclusions	141
7.2.	Future work.....	143
7.3.	Publications and projects	145
7.3.1.	List of publication	145
7.3.2.	Research projects.....	146
Appendix A: Ontology descriptions		147
Appendix B: Reasoning APIs.....		158
References		161

List of figures

Figure 1. Context awareness lifecycle.....	3
Figure 2. The SWARMs architecture.....	5
Figure 3. Different uncertainty annotations.	16
Figure 4. PR-OWL simple model.	21
Figure 5. Mapping of PR-OWL random variables and OWL properties.	22
Figure 6. Semantic and context-aware framework for autonomous underwater vehicles.	27
Figure 7. OODA loop for processing data.	27
Figure 8. Context awareness concepts.	28
Figure 9. The architecture of CAMPH.	29
Figure 10. ACoMS+ architecture.....	30
Figure 11. CA4IOT architecture.	32
Figure 12. The architecture of CAMPUS.....	34
Figure 13. The architecture of the CASF.....	35
Figure 14. SeCoMan, as described in [108].	36
Figure 15. The FlexRFID framework.....	39
Figure 16. Overview of the proposed context-aware framework.....	47
Figure 17. Overview of the fuzzy ontology development methodology.	52
Figure 18. The flow of step 3, 4, and 5.....	54
Figure 19. UMP-ST, extracted from [126].	61
Figure 20. POMC, extracted from [126].	62
Figure 21. The general architecture of the SWARMs ontology.....	66
Figure 22. A representation of the overall structure of the core ontology.....	68

Figure 23. Robotic vehicle taxonomy.	69
Figure 24. The mission & planning ontology.	70
Figure 25. The environment recognition & sensing ontology.	72
Figure 26. Communication & networking ontology.	73
Figure 27. Graphical user interface provided by Protégé.	74
Figure 28. The snapshot of the hierarchy of the SWARMs ontology.	75
Figure 29. Fuzzy data types for seabed region size.	78
Figure 30. The overall visualized structure of the seabed characterization fuzzy ontology.	78
Figure 31. Entities, their attributes, and relations for the oil spill monitoring model.	81
Figure 32. The oil spill monitoring probabilistic ontology formatted in an MEBN theory.	82
Figure 33. MFrag Thickness.	84
Figure 34. LPD for Thickness(oil).	84
Figure 35. MFrag EstimatedSize.	84
Figure 36. LPD for EstimatedSize.	84
Figure 37. MFrag Weather.	85
Figure 38. LPD for Weather.	85
Figure 39. MFrag Currents.	85
Figure 40. LPD for Currents.	86
Figure 41. MFrag Location.	86
Figure 42. MFrag SpreadSpeed.	87
Figure 43. LPD for SpreadSpeed.	87
Figure 44. The LPD of SpreadSpeed represented in the PR-OWL syntax.	88
Figure 45. MFrag SeverityLevel.	88
Figure 46. LPD for SeverityLevel.	89

Figure 47. SROM-based uncertainty propagation in ontologies.	94
Figure 48. The distribution of 10^4 instances of Length and Radius.	97
Figure 49. CDF of Volume obtained by SROM-based method ($m=20$) and benchmark.	98
Figure 50. CDF of Volume obtained by SROM-based method and benchmark.	99
Figure 51. Error of mean of Volume.	100
Figure 52. Error of Standard deviation of Volume.	100
Figure 53. The hybrid context reasoning mechanism.	104
Figure 54. Applying the Pellet reasoner to facilitate ontological reasoning.	109
Figure 55. An example of ontology inconsistency in the SWARMs ontology.	110
Figure 56. The newly inferred knowledge shown in Protégé.	111
Figure 57. The RobotCandidateForMission rule in Protégé.	112
Figure 58. The rule-based reasoning result in Protégé.	113
Figure 59. Findings in this scenario.	114
Figure 60. The MEBN theory shown in the UnBBayes GUI.	115
Figure 61. Creation of an instance of Currents.	115
Figure 62. SSBN generated for the scenario with complete findings.	116
Figure 63. SSBN generated for the scenario with incomplete findings.	117
Figure 64. Implementation scope within the semantic middleware.	119
Figure 65. The SWARMs ontology metrics.	120
Figure 66. The use of the ontological reasoning.	125
Figure 67. The first kind of use of the rule-based reasoning.	126
Figure 68. The second kind of use of the rule-based reasoning.	127
Figure 69. The use of the MEBN reasoning.	128
Figure 70. Component diagrams.	129
Figure 71. Ontological reasoning test.	131

Figure 72. Comparison of ontological reasoning time with different individuals.	133
Figure 73. Rule-based reasoning test.	133
Figure 74. Query result from the SWARMS ontology after ontological reasoning.	135
Figure 75. Query result from the SWARMS ontology after rule-based reasoning.	135
Figure 76. The performance of rule-based reasoning.	136
Figure 77. The query result of the severity level of region_1.	137
Figure 78. The performance of the MEBN reasoning.	138
Figure 79. APIs of the OntologicalReasoner class.	158
Figure 80. APIs of the RuleBasedReasoner class.	159
Figure 81. APIs of the MEBNReasoner class.	160

List of tables

Table 1. Comparison of existing context modeling methods	13
Table 2. Fuzzy data types	76
Table 3. Fuzzy concepts	77
Table 4. Comparisons between SROM-based results and reference	101
Table 5. Partial ontological reasoning constructs	105
Table 6. ontologicalReasoning interface	129
Table 7. ruleBasedReasoning interface	129
Table 8. mebnReasoning interface	130
Table 9. Concepts defined in the robotic vehicle ontology	147
Table 10. Relations defined in the robotic vehicle ontology	148
Table 11. Data properties defined in the robotic vehicle ontology	148
Table 12. Concepts defined in the mission & planning ontology	149
Table 13. Object properties defined in the mission & planning ontology	150
Table 14. Data properties defined in the mission & planning ontology	151
Table 15. Concepts defined in the environment recognition & sensing ontology	151
Table 16. Object properties defined in the environment recognition & sensing ontology	153
Table 17. Data properties defined in the environment recognition & sensing ontology	155
Table 18. Concepts defined in the communication & networking ontology	156
Table 19. Object properties defined in the communication & networking ontology	157
Table 20. Data properties defined in the communication & sensing ontology	157

List of equations

(1).....	91
(2).....	91
(3).....	91
(4).....	91
(5).....	91
(6).....	91
(7).....	92
(8).....	92
(9).....	93
(10).....	93
(11).....	93
(12).....	93
(13).....	93
(14).....	96

1. Introduction

1.1. Context awareness

Context awareness is continually obtaining interests in many research domains [1], such as smart cities, smart homes, ambient assisted living, and smart grid. As a key enabler for entities to understand their environment and make adaptations accordingly, context awareness implies an effective exploitation of context. The term context awareness is often referred to be sentient, reactive, context-sensitive, environment-oriented, situated, responsive, and adaptive. It firstly appeared in the Active Badge research object of Olivetti Research Ltd. (Cambridge, England) in 1992 [2]. Since then, this term has been generalized with different definitions, e.g.,

- *“...the ability of computing devices to detect and sense, interpret and respond to aspects of a user’s local environment and the computing devices themselves.” [3]*
- *“...limited to the human-computer interface [4], and the notion of adaptation [5].”*
- *“...provide the maximum flexibility of service based on real-time context.” [6]*
- *“...automatically provide information or take actions according to the user’s present context and need.” [7]*
- *“...if an application has the ability to monitor input from sensing devices and choose the suitable context according to user’s need or interests, then it can be labeled as a context-aware application.” [8]*
- *“...a system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task.” [9]*

The last explanation for context awareness is considered as the most general definition and it is acknowledged throughout this thesis. With advances in sensing, pervasive computing and communication technologies, more and more contexts can be obtained and promisingly used. To make the most of the available context is a key factor to achieve context awareness.

To achieve context awareness, it is necessary to obey a general rule. Such a rule is called context lifecycle [1]. The management of context, which is the period of time from its obtainment to destruction, is demarcated by five significant events as Context Acquisition, Context Modeling, Context Reasoning, Context distribution, and Context Repository as illustrated in Figure 1. The context awareness cycle begins with the acquisition of various kinds of context followed by the formalization and inference process and finally ends up with the distribution of useful context to the corresponding applications. At the stage of context modeling and reasoning, historical context data needs to be recorded for further use or queries.

Figure 1. Context awareness lifecycle.

Specifically, context acquisition aims to obtain a maximum amount of context data. Context can be generally classified as physical context (obtained from sensing devices selected according to requirements a certain application or system) or virtual context (not sensed, but obtained in other ways, such as provided by users, or derived from existing context). In this sense, different kinds of context data which might be structured in different formats can be obtained in the acquisition phase. To make use of them, the premise is to define and store it in a machine readable and processable form, hereby all data should be converted into a unified format such that the context can be understood and shared. This can be achieved by a model that defines, represents and processes the object “context”. This phase is called context modeling. Based on the context model, context reasoning aims to derive high-level context which contains more meaning. Context distribution is responsible for disseminating useful context information to corresponding applications. Two typical distribution mechanisms, subscribe/publish and polling, are widely used in current solutions [10].

In general, three typical approaches have been of much value to facilitate context awareness in applications [11]: 1) each application interacts, obtains, processes, and uses the context of its interest in its own manner; 2) some libraries/toolkits aiming at acquiring and processing context can be added and reused for building context-aware applications; and 3) a context-aware framework/middleware/intermediation architecture is adopted to provide common functionalities to manage context and deliver context awareness. The third approach is regarded as the best solution due to its ability to decrease the complexity of building context-aware applications [12]. Hence, context-aware frameworks are highlighted as an essential requirement for facilitating context awareness. Context-aware frameworks should provide a fundamental context management which complies with the context awareness lifecycle. As a result, by using context-aware frameworks, developers can be freed from the concern of managing the context and they can focus on designing desired application functions and business logic.

1.2. The importance of context awareness in the cooperation of underwater robots

Underwater robots are gaining momentum as useful tools when issues and challenges spring up regarding maritime operations. Infrastructure maintenance [13], oil spill counter measurements [14], and survey operations [15] are only a few of the use cases that these devices have proven to have a major usability in. Underwater robots consist of heterogeneous vehicles capable of providing different functionalities, such as Autonomous Underwater Vehicles (AUVs), Autonomous Surface Vehicles (ASVs), and Remotely Operated Vehicles (ROVs), to perform maritime and underwater-related tasks. While dealing with complex missions that outmatch the capabilities of a single robotic vehicle (e.g., it is not realistic to use a single ROV for constructing a berm on the seabed) or operations that are dangerous for divers, seamless cooperation between different vehicles is demanded to tackle the complexity or riskiness of maritime missions.

Knowing and understanding the physical environment where specific operations are to take place is key for accomplishing effectively and efficiently such operations and associated activities. Nonetheless, it is very difficult or nearly impossible to know such environment with an uncertain and dynamic nature a priori or even in a rough way [16]. In particular, when the considered activities for underwater robots become more complex, there is a demanding need to explore approaches targeting high efficiency and exploiting cooperation in a highly dynamic setup. Enabling underwater robots to be context-aware is a promising approach to achieve the potential cooperation between robots in such complex activities [17]. The context awareness feature can endow robots with the capability of gathering, exchanging, federating, analyzing, reasoning, and acting on available context information. It can enable heterogeneous robots to achieve a common understanding of information exchanged between them and provide possibilities for robots to cooperate in a better way. In addition, being context-aware, robots can know the big picture of the environment by making effective use of available context information and further adapt their behaviors accordingly to any change occurred in the environment. This implies a great potential for context-aware robots to accommodate to the dynamic environment and carry out complex missions.

1.3. Research framework

Introducing context awareness into the field of underwater robots is still under-researched. In current literature, only a few attempts, including Pandora [18], Trident [19], and CoCoRo [20],

have been presented for delivering context awareness in underwater robots. However, they differ from each other in terms of context awareness level and they present unsatisfactory performance due to the lack of important data treatments.

Based on this context, a new European project, Smart and Networking Underwater Robots in Cooperation Meshes (SWARMS) [21], has been initiated and it includes introducing context awareness into the field of underwater robotics as one of its objectives. This project aims to expand the use of unmanned underwater vehicles (e.g., AUVs or ROVs) in maritime and offshore operations in a collaborative, cooperative, and context-aware manner. Figure 2 shows the general architecture that is currently being considered in the SWARMS project. It involves three main functional components: a **mission management tool (MMT)**, the **middleware** layer, and **robot systems (RS)**. The MMT is physically implemented in a command and control station (that could be on land or in a vessel). The set of robotic systems (hardware and software components) involved in a given mission scenario could be addressed with its corresponding functional components too. However, the middleware layer is distributed among various physical components by nature (e.g., the command and control station and the robotic systems).

Figure 2. The SWARMs architecture

To summarize, the middleware system ensures the collaboration between robots and other physical components of the system in order to achieve the tasks that the mission management tool outputs when carrying out a mission. In general terms, the different goals of a mission, the tasks in which they are sub-divided, and the relationships among all them are implemented in the command and control station and somehow translated into tasks that could be more or less specific and that are transmitted through the middleware to the robotic systems. The robotic systems then must collaborate in different degrees to carry out those tasks. The translation of tasks implemented in the command and control station into commands that the middleware actually understands, and then again translated into different specific robotic systems commands, are carried out by different parts of the middleware that could be implemented in different physical components.

This thesis is carried out in the context of the SWARMs project. Motivated by one of the goals defined in the SWARMs project, it aims to find solutions for facilitating context awareness in the cooperation of underwater robots so that robots can know and understand the physical environment where specific operations are to take place and also be able to adapt to any change occurred in the dynamic underwater environment.

1.4. Problem statement and thesis objectives

To deliver context awareness in the cooperation of underwater robots is the ultimate objective of this thesis. However, it is not an easy task to achieve this goal due to the problems identified as follows:

Problem 1: The context awareness feature should be used to provide a complete picture of the environment in question, as well as the current conditions, to entities (e.g., robots, operators, MMT) so that they can take optimized decisions with regard to required actions in a SWARMs mission, such as requesting robotic vehicles to move to certain locations. The context awareness feature must be able to be exploited in many ways, such as enhancing communications, improving vehicle operations, or optimizing mission planning. For that, context awareness needs to be taken into consideration right at the design phase of the SWARMs general architecture and also in the definition of system requirements. How to provide a general context awareness solution for the SWARMs robots and also enable the context awareness solution customized to the robot field remains a challenge.

Problem 2: In the underwater robotic field, a variety of information could be obtained from different data producers, such as sensors, vehicles, and external sources (e.g., weather forecast

system). Substantial difficulties arise when attempting to exchange and use information between vehicles. In particular, data heterogeneity obstructs effective information exchange between vehicles to a great extent. Information, pertained to heterogeneous data producers, may be described in different formats, e.g., Extensible Markup Language (XML) [22], JavaScript Object Notation (JSON) [23], depending on their type and involved manufacturer. In addition, even when different vehicles use the same terminology, sometimes it is interpreted with different meanings. For instance, the term Position is used to represent a local frame georeferenced location in robot A while robot B uses it to express its angular coordinates. Vehicles need to have a common understanding of context information that is exchanged and shared among themselves before further using them. In addition, context uncertainty is the norm rather than the exception in the underwater robot field due to a variety of factors, e.g., imperfect instruments and unreliable communications. In addition to developing a formal representation of context information in this domain, there is an increasing need to develop a formal characterization of the uncertainty associated with attributes of the context information. In other words, focusing on the context modeling phase, how to model heterogeneous context information and its uncertainty in a formal and comprehensive manner must be resolved.

Problem 3: High-level context information, derived from available context information, can be more useful for vehicles or MMT to understand the operational environment or conditions. For instance, a piece of high-level context information, *vehicle A might collide with vehicle B soon*, is more meaningful for operators to make decisions than basic context information *vehicle A is out of its assigned trajectory and heading in the direction of vehicle B*. Therefore, how to make effective inference on available heterogeneous context information is worth to be studied. In addition, since context uncertainty is pointed out in problem 2 as an inherent characteristic of context information obtained in the underwater robot field. Making inferences over uncertain information reveals another difficulty. In summary, it is necessary to find appropriate context reasoning methods to meet reasoning requirements exposed in the SWARMs project.

Aiming to tackle the problems identified before, the main goal of this thesis, that is providing a solution to deliver context awareness in the cooperation of underwater robots, is subdivided into the following sub-goals.

- A comprehensive study on the state of the art, including:
 - Existing technologies
 - Selection of technologies appropriate for this thesis

- Design of a general solution to facilitate context awareness based on the following technologies:
 - Context-aware framework
 - Component-based architecture
- Modeling information related to underwater robots by using
 - Networked ontologies
 - Fuzzy logic based extensions, namely, fuzzy ontologies
 - Probabilistic logic based extensions, namely, probabilistic ontologies based on Probabilistic OWL (PR-OWL) which complies with the Multi-Entity Bayesian Network (MEBN) theory
- A hybrid context reasoning mechanism for providing effective reasoning about modeled information, including
 - Ontological reasoning method
 - Semantic Web Rule Language (SWRL) rule-based reasoning method
 - MEBN-based reasoning method
- Implementation and validation, including
 - Implementing the information model to represent information in the cooperation of underwater robots
 - Implementing the hybrid context reasoning mechanism
 - Validating the reasoning proposal based on the information model
 - Scenario selection
 - Preliminary quantitative performance analysis
 - Results

1.5. Research contributions and thesis structure

This thesis makes a set of contributions that can be summarized as follows:

- A context-aware framework proposal to provide a general solution for delivering context awareness by providing a complete and well-defined context management.
- A new Fuzzy Ontology Development Methodology (FODM) proposal that presents the first methodological guide for building fuzzy ontology [24] from scratch.
- An ontology proposal that is able to provide a formal and comprehensive representation for context information and associated uncertainty based on fuzzy logic and Multi-Entity Bayesian Network (MEBN) [25] theories and provide support for uncertainty reasoning.

- A proposal of applying the Stochastic Random Order Model (SROM) [26] to quantify uncertainties propagated in mathematic relationships in ontologies.
- A hybrid context reasoning mechanism proposal that integrates the ontological, rule-based, and Multi-Entity Bayesian Network (MEBN) reasoning to provide different reasoning capabilities in order to accommodate different reasoning requirements.

The remainder of this thesis is organized as follows:

- Chapter 2 introduces existing work related to the topic of this thesis and also discusses the techniques that are employed in this thesis.
- Chapter 3 presents an architectural proposal of a context-aware framework for being used in the cooperation of underwater robots. Functionalities of different components involved in the context-aware framework are described.
- Chapter 4 presents the contributions made by this thesis to the context modeling field. Firstly, it reviews existing ontology development methodologies and proposes a new fuzzy ontology development methodology. Afterward, an ontology proposal, named the SWARMs ontology, is described in detail. Finally, focusing on mathematic relationships in ontologies, a proposal of applying the Stochastic Random Order Model (SROM) to quantify uncertainties in such relationships is presented.
- Chapter 5 describes a hybrid context reasoning mechanism which includes three different context reasoning methods in the cooperation of underwater robots. A preliminary analysis on the capabilities of the adopted context reasoning methods is also provided.
- Chapter 6 shows the detailed implementations for the proposed SWARMs ontology and the proposed context reasoning mechanism. The usefulness and performance of different context reasoning are validated.
- Finally, Chapter 7 provides a summary of the major contributions of this thesis. In addition, some future work recommendations for this research topic are provided.

2. Background and related work

2.1. Context modeling

2.1.1. Existing context modeling methods

Ten modeling techniques, including “Key-value”, “Markup”, “Graphical”, “Object-oriented”, “Logic-based”, “Multidisciplinary”, “Domain-focused”, “User-centric”, “Ontology-based”, and “Chemistry-inspired”, are overviewed in this section. The fundamental scheme to examine the available context modeling techniques is based on the data structure used for representation.

Key-Value Context Modeling

Key-value pairs are used to enumerate attributes and values in this model. The model can be written in different formats (e.g., text and binary). Because of its simplicity and ease of use, it was widely employed in early research and various service frameworks. For example, Schilit [27] describes the limited number of location information as key-value pairs. However, it lacks capabilities for complex structuring for enabling efficient context retrieval algorithms.

Markup Context Modeling

This is referred to as tagged encoding, as context information is stored within tags, *i.e.*, symbols and annotations which represent and format the data. Those symbols and annotations originate from typical markup languages such as XML. Typical representatives of this model are profiles. The limit of this model is that its hierarchical structure should be pre-defined and also it is useless to capture context relationships.

Graphical Context Modeling

Graphical diagrams enabled by this model are able to specify relationships. Three widely used model examples are Unified Modeling Language (UML) [28], Entity Relationship Model (ERM) [29], and Object Role Model (ORM) [30]. The UML is a standardized general-purpose language which expresses different kinds of context information in an own graphical notation, whereas ERM and ORM work for designing and querying databases at the conceptual level. However, the interoperability among different storage databases which are used in the actual low level of graphical model poses a challenge.

Object-Oriented Modeling

The object-oriented model [31] employs class hierarchies and relationships to represent context data and incorporates encapsulation, inheritance, and reusability into context expression. Instances can be allowed to access the context by inheritance mechanisms. The core component

is called entity and it forms the subject of structured context information. An entity is linked to other entities by means of attributes which are also called associations. This technique stresses developers in terms of being aware of the whole context taxonomy.

Logic-Based Context Modeling

In a logic-based model, context is defined as facts, expressions, and rules. It is flexible to add, update or remove data in this model. This model thus offers a high degree of formality. A variety of applications has adopted this model. e.g., in [32] a model in a seven field data structure (subject, predicate, object, time, area, certainty, freshness) is developed which helps to organize the information in a sequence. This model is an enabling method to check context consistency and to support the reasoning task as well. However, it still lacks standards and validation tools.

Multidisciplinary Context Modeling

This model involves, as the name says, multiple disciplines like psychology, computer science, and linguistics [33]. It demonstrates context from different points of view and specifies the relationships among multiple disciplines. The idea is to widen the vision to examine the context and to construct a general model, but the complex modeling process introduces difficulties as it incorporates the information concerning many applications, various types of users, and multiple environments. This proposal still remains at the conceptual level. The specific procedures are not clearly figured out and thus the practical usage of this technique is rare.

Domain-Focused Context Modeling

Domain-focused context modeling, also referred to as W4 context model, is tailored to model an application domain. Therefore, Castelli *et al.* [34] elaborates the specific mechanism as a four fields tuple [Who, What, Where, When] (the elements are also called knowledge atoms) that is recognized in “Someone or something (who) does/did some activity (What) in a certain place (Where) at a specific time (When)”. This model is very expressive and flexible for data usages. Queries, modification, and deletion are allowed on context tuples.

User-Centric Context Modeling

As the name says, this model is built from a user’s perspective and explores how context information is perceived by users instead of devices, services or applications [35]. Here, “How” and “Why” is added to the formerly presented W4-tuple and leads to the 5W1H-tuple: [Who, When, Where, What, How, Why]. More details about this model are presented in [35]. This method

can express context in a very organized way, however, it is just a trade-off between the complexity of expression and ease of use.

Chemistry Inspired Context Modeling

Ikram *et al.* [36] explored similarity between chemistry and context modeling to fully use chemical reactions and periodic table representation. The general scheme is that context is represented in a reactive model called “Smart Space” [37] [38] where associated services can be triggered like chemical bonding or chemical reactions. It is capable of representing various kinds of context and invoking right reactions automatically. However, it is still known by just a few people due to its infancy. Besides, it is difficult for the Smart Space model and corresponding graphical visualization to dynamically evolve when the amount and type of context grow.

Ontology-Based Context Modeling

Studer [39] defined ontology as “...a formal, explicit specification of a shared conceptualization” in the semantic field. Concepts, instances, and relationships (as main components of ontology) can formally and comprehensively represent the knowledge. One of the main advantages of using ontologies is their way to represent and share knowledge by using a common vocabulary. They formulate and model relationships between concepts in a given domain. As providers of format for exchanging knowledge, they promote interoperability, knowledge reuse, and information integration with automatic validation. This modeling method is regarded to as the most promising method in [1] [40] and it can address the conceptual confusion among people and systems because it shares the common understanding. Different formalism languages, including RDF (Resource Description Framework) [41], RDFS (Resource Description Framework Schema) [42], and OWL (Web Ontology Language) [43], can be used to formalize ontology in a machine-readable format. Ontology is competitive over other models in terms of interoperability, formality, and reusability.

Table 1 summarizes the most important features of every technique such as advantages and disadvantages.

Table 1. Comparison of existing context modeling methods

Context Modeling Technique	Advantages	Disadvantages	Applicability
Key-value	Simple; Ease of use; Flexible	Lack of standards; Useless when big in size; Cannot	Adequate to model limited context in simple and self-independent

		represent relationships; Difficult to retrieve information; Lack of validation tools; Lack of scalability; Only exact matching.	applications which do not need to share data with other applications. Examples: [27] [44]
Markup	Structured; Some validation tools are available; Flexible.	Lack of standards; Problems in capturing relationships; Timeless; Dependencies; Inconsistency checking; Reasoning and Uncertainty.	Efficient as a mode of data transfer about shallow context over the network; Applications in which levels of information are few. Examples: [45] [46]
Graphical	Rich expressiveness; Relationships are allowed; Validation is possible through constraints; Different standards and implementations are available.	Interoperability is unsolved; Configuration must be required; A generic and well- developed standard is needed.	Particularly applicable to derive an ER-model which is useful as structuring instrument for a relational database. Examples: [47] [48]
Object-oriented	Relationships are allowed; Some development tools are available; Can be fused by using programming languages.	Lack of standards; Lack of validation; Hard to retrieve information; Reasoning is not supported.	Suitable to be used in code-based (high-level programming languages) applications with high computational capability. Examples: [31] [49]
Logic-based	Rich expressiveness; Support reasoning; Consistency check; Simplicity; Processing tools are available.	Lack of standards; Lack of validation.	Suitable for applications in which high-level information is needed and developers are willing to specify constraints. Examples: [32] [50] [51]
Multidisciplinary	Comprehensive understanding of context based on multiple disciplines; The division of context is concrete.	Too complex; Still at the first stage; Interoperability is unsolved.	Tailored to applications in which key human and social issues should be identified. Examples: [33]
Domain-focused	Expressive; Flexible; Structured.	Lack of standards; Lack of validation.	Suitable to single domain-focused applications. Examples: [34]
User-centric	Express context in an organized way; Scalability; Allow reasoning.	Lack of standards; Complex to use; Lack of validation; Lack of formality.	Suits applications focused on perspectives of users; data expression is in an intuitive manner. Examples: [35]

Chemistry inspired	Medium expressivity to represent many kinds of context; Support for triggering services autonomously; Cross-domain inspired.	Lack of standards; Lack of validation; Not dynamic and scalable; In a nascent stage.	It is possible to apply this model to applications which require spontaneous interaction and composition of information. Examples: [36]
Ontology-based	Support reasoning; Rich expressiveness; Relationships are allowed; Strong validation; Processing tools available; Mature standards; Interoperability.	Representation can be complicated; It will be complex to retrieve context information; Unable to address uncertainty.	Suitable for applications which highly need to exchange information with others; Sufficient knowledge engineering skills are available. Examples: [39] [52] [53] [54]

Recalling the introduction for each modeling technique and observing Table 1, it seems obvious that none of the methods is ideal as a standalone technique because all have some limitations. In addition, a noteworthy criterion to choose an appropriate context modeling language is if it can provide solutions in context reasoning by seeking for capturing a variety of context types along with relationships, dependencies, and quality of content. Some of them are only applicable in very simple applications such as key-value and markup methods. The usage of multidisciplinary and chemistry inspired models is considerably limited as they are emerging techniques lacking theoretical support. For the rest of the methods, including graphical, object-oriented, logic-based, domain-focused, and user-centric, drawbacks such as lack of interoperability and complexity prevent them from being widely used. The comparison leads to the conclusion that the ontology-based context modeling method could be the most promising technique to model context, because many technical obstacles like interoperability, support for reasoning, strong validation, and expressivity are overcome by adopting ontology. However, a fact, that classical ontology is not appropriate to deal with uncertainty, which is inherent to context obtained in most of the real applications [55], should not be negligible. Therefore, there might not be a single context modeling technique to be used in a standalone fashion. Perera *et al.* [1] defended that the best way to model context is to create a novel technique to integrate the existing context modeling techniques. For example, Liang and Cao [56] introduced a hybrid model that combines graphical and ontological techniques, while Bobillo and Straccia [24] proposes the integration of fuzzy logic with ontology so as to advance the classical ontology to fuzzy ontology.

2.1.2. Uncertainty modeling

The uncertainty inherent to data could be varied depending on different applications. Therefore, different understanding of the uncertainty of ontology representation coexists to fit specific interests of applications. The W3C UR3W-XG [57] group proposed an uncertainty ontology that captures top-level classes and properties for characterizing the uncertainty consideration in ontologies. According to this uncertainty ontology, uncertainty can be classified into five main types, namely, *Ambiguity*, *Randomness*, *Vagueness*, *Inconsistency*, and *Incompleteness* [58]. The uncertainty concern in ontologies comes from mainly two kinds of derivation: objective and subjective.

Figure 3. Different uncertainty annotations.

Ambiguity: The referents of terms in a sentence to the world are not clearly specified and therefore it cannot be determined whether the sentence is satisfied.

Vagueness: This uncertainty feature indicates that there is not a precise correspondence between terms in the sentence and referents in the world. For instance, the seabed size is large with a degree of 0.6.

Inaccuracy: Real-world information could be inaccurate.

Incompleteness: It implies that information about the world is incomplete, some information is missing.

Randomness: Sentence is an instance of a class for which there is a statistical law governing whether instances are satisfied.

As emphasized in section 2.1.1, a classical crisp ontology cannot represent context uncertainty in real-world applications. Fortunately, fuzzy and probabilistic logic, shown in Figure 3, have proven to be suitable formalisms to handle ambiguity & vagueness and inaccuracy &

incompleteness & vagueness, respectively. Several attempts have been followed. For instance, it is common to use simple XML tags to express the probability value which is a number ranging from 0 to 1. However, applying these simple probability annotations fails to convey the structure of the probabilistic representation. Probability has been regarded to be more about structure than it is about numbers [59]. Thus, a principled uncertainty modeling is preferred. For instance, to model uncertainty from a probabilistic aspect and enable the ease of applying Bayesian Network (BN) reasoning [60], Yang *et al.* [61] proposed a probability-annotation approach. Specifically, three ontology concepts, including "PriorProb", "CondProb", and "FullProbDist", were defined to conceptualize collections of Bayesian related definitions. In addition, a data property "ProbValue" was proposed to link "PriorProb" and "CondProb" with the probabilistic value varying from 0 to 1. Two emerging modeling methods, fuzzy ontology and probabilistic ontology, are detailed in the following.

Fuzzy Ontology

The nature of much real-world information is imprecise, vague, and ambiguous. Crisp ontologies cannot handle this type of information since they can only model relations between entities that may be either true or false. For instance, in the statement "*AUV alister is repairing the pipeline*", the activity repairing can be recognized with some degree of truth depending on the sensor data acquired and how the robot is performing the activity. Bobillo *et al.* [24] [62] introduced fuzzy logic in crisp ontologies to capture and represent such vagueness and ambiguity. Different from classical set theory, where elements either belong to a set or not, elements in the fuzzy set theory can belong to a set with some degree. Formally, a membership function $\mu_A(x)$ or simply $A(x)$ can define a fuzzy subset A of X and it can assign any $x \in X$ to a value in the real interval between 0 and 1. Approximate reasoning involving inference rules with premises, consequences, or both of them containing fuzzy propositions can be enabled by fuzzy logic [63]. More details of definitions for fuzzy ontology can be found in [62] [64]. The most general definition for fuzzy ontology is ***an ontology that uses fuzzy logic to provide a natural representation of imprecise and vague knowledge and ease reasoning over it*** [64]. Essentially, elements which form fuzzy ontologies are similar to those in crisp ontologies from the definition point of view. An exhaustive list of fuzzy ontology elements can be referred to [24]. The most widely elements included in fuzzy ontologies are shown in the following:

- *Fuzzy concepts.* They refer to concepts which do not have clear-cut boundaries and represent fuzzy sets of individuals. Thus, an individual could be attributed to a fuzzy concept with a certain degree. For instance, Jack aged 45 could be classified as an

instance of a fuzzy concept *YoungPerson* with a degree of 0.4. So instead of being impossible, Jack is regarded as a young person to some extent.

- *Fuzzy roles/object properties.* Fuzzy roles describe fuzzy binary relations between concepts or individuals. They can link different concept instances associated with certain degrees. For instance, a fuzzy relationship "likes" can be used to represent a vague statement "*John likes apples to degree 0.8*".
- *Fuzzy data types.* Fuzzy forms of data which contain vague meanings are specified by fuzzy data types. Fuzzy data types are used to fuzzify attributes values, such as the range of data properties.

Different ways to represent fuzzy ontologies can be found in [64] [65]. However, the methodology, proposed by Bobillo and Straccia [24], to formalize fuzzy ontologies using OWL 2 annotation properties is considered as the most convenient way in this thesis. In addition, the fuzzy Description Logic (fuzzyDL) [63] [66] reasoner is also available for reasoning over fuzzy ontologies.

Probabilistic Ontology

A probabilistic ontology is an explicit, formal knowledge representation that expresses knowledge about a domain of application [67]. This representation includes:

- types of entities existing in the domain;
- properties of those entities;
- relationships among entities;
- processes and events that happen with those entities;
- statistical regularities that characterize the domain;
- inconclusive, ambiguous, incomplete, unreliable, and dissonant knowledge;
- uncertainty about all the above forms of knowledge;

where the term entity means any concept (real or fictitious, concrete or abstract) that can be described and reasoned about within the domain of application.

In essence, probabilistic ontologies aim to comprehensively describe knowledge about a domain and the uncertainty associated with that knowledge in a probabilistic logic way. Probabilistic Web Ontology Language (PR-OWL) [59] is a language to represent probabilistic ontologies. PR-OWL can also be regarded as an OWL upper ontology for representing uncertainty that relies on Multi-Entity Bayesian Networks (MEBN) [68]. Based on the MEBN logic, PR-OWL can extend the capability of OWL ontologies to represent complex Bayesian probabilistic models

and provide fundamental support for probabilistic reasoning. In the following, the MEBN and PR-OWL principles will be briefly described.

Multi-Entity Bayesian Network (MEBN)

It is known that Bayesian Network (BN) is effective to model and reason a fixed number of hypotheses. However, it cannot reason problems which involve a varying number of entities. To address this shortcoming of BN, Multi-Entity Bayesian Networks (MEBN) [68], as an extension of standard BN, was proposed by Laskey *et al.* MEBN extend BN to achieve first order expressive power. With the incorporation of first order logic and BN, the MEBN is able to provide a consistent treatment of uncertainty, such as representing uncertainty about the type of an entity, refining type-specific probability distributions through Bayesian Learning, and reasoning under uncertainty. Beyond the capability of traditional BN in reasoning about a fixed number of attributes, MEBN could deal with a varying number of entities with their number, type, and relationships undetermined. MEBN provides a means of defining probability distributions over an unbounded and varying number of interrelated hypotheses with the aid of syntax, a set of model construction and inference processes, and semantics.

MEBN interprets the world as a set of entities that have attributes and have causal relationships with other entities. Knowledge about the attributes of the entities and their relationships to each other is represented as an MEBN model. MEBN logic consists of a collection of MEBN fragments (MFrag) organized into an MEBN Theory (MTheory). An MTheory could represent a particular domain of discourse. Each MFrag, as a modular component, represents knowledge about specific subjects within the domain of discourse and models probability information about a group of related random variables. Similar to BN, each MFrag is a Directed Acyclic Graph (DAG) [69] with parameterized nodes that represent attributes of entity and edges that represent dependencies among them. More specifically, an MFrag contains random variables (RVs) and a fragment graph representing dependencies among these RVs. Each MFrag represents a repeatable pattern of knowledge that can be instantiated as many times as needed to form a BN addressing a specific situation, and thus can be seen as a template for building and combining fragments of a Bayesian network. It is instantiated by binding its arguments to domain entity identifiers to create instances of its RVs. Three types of random nodes, resident nodes, context nodes, and input nodes, are defined in the MTheory.

- *Resident nodes.* In an MFrag graph, resident nodes represent variables that have local probability distributions dependent on the values of their parents. Exactly one home MFrag is assigned to contain the complete expression of a resident node.

- *Context nodes.* Context nodes are Boolean nodes, including value *True*, *False*, and *Absurd*. The MFrag must satisfy the conditions expressed by context nodes in order to be valid.
- *Input nodes.* Input nodes have their distributions defined in other MFrams and they are important inputs for the definition of resident nodes.

A set of MFrams represents a joint distribution over instances of its random variables. MEBN enables a compact way to represent repeated structures in a BN. The major convenience provided by MEBN is that there is no fixed limit on the number of RV instances, and the random variable instances can be dynamically instantiated to accommodate different scenarios.

To apply an MTheory to reason about specific scenarios, there is a need to provide the system with specific information about the individual entity instances involved in the scenario. Based on the provided information and a given inference query, the MEBN model can be instantiated, based on Laskey's algorithm, and generate a standard BN, which is called Situation Specific Bayesian Network (SSBN) [68]. Therefore, a standard Bayesian inference can be used both to answer specific questions of interest (e.g., how likely the two robots may collide?) and to refine the MTheory. Essentially, the Bayesian inference is applied to perform both problem specific inference and learning from data in a sound way.

Probabilistic OWL (PR-OWL)

Probabilistic OWL (PR-OWL) [59] [67] is a language for representing probabilistic ontologies. The uncertainty annotations provided by PR-OWL comply with the MEBN logic. In this way, PR-OWL can not only provide a consistent representation of uncertain knowledge that can be reused by different probabilistic applications but also allow applications to perform probabilistic reasoning with that knowledge. PR-OWL provides a means of expressing subtle features required to express MEBN theories. More specifically, it consists of a set of classes, subclasses, and properties that collectively form a framework for building probabilistic ontologies. PR-OWL is an extension to the OWL language based on MEBN, thus, it is interoperable with non-probabilistic ontologies. The PR-OWL or the probabilistic definitions from an OWL ontology merged with PR-OWL have to form a valid complete or partial MTheory. The main concepts involved in defining an MTheory in PR-OWL can be seen in Figure 4. In this figure, ellipses represent general classes and their relationships are represented using arrows. A probabilistic ontology must have at least one individual of class MTheory which includes several MFrams. In actual PR-OWL syntax, the relationship between class MTheory and class MFrag is specified by an object property *hasMFrag*.

Individuals of class MFrag are formed by nodes. Each individual of class Node is a random variable and it has a mutually exclusive and collectively exhaustive set of possible states. These possible states are individuals of class Entity. Random variables have unconditional or conditional probability distributions which are expressed by class ProbabilityDistribution. The more detailed PR-OWL syntax can be seen in Figure 5. For instance, the relationship between PR-OWL random variables and OWL properties is formalized using the relation *definesUncertaintyOf*. Besides, the relation *definesUncertaintyOf* can be used to relate the PR-OWL random variable *isObjectOfInterest* (Entity) to the OWL property *isObjectOfInterest*. Properties *isSubjectIn* and *isObjectIn* are defined to link arguments of the random variables with their OWL properties depending on whether they refer to the domain or range of the OWL property. The full details on the PR-OWL ontology can be found in [67]. Due to the complexity of PR-OWL syntax, the building of probabilistic ontologies using PR-OWL in protégé is a manual, error-prone, and tedious process that requires a thorough understanding of the logic and the syntax of PR-OWL. Fortunately, the UnBBayes [70] [71] [72] tool provides a GUI-based editing process to create probabilistic ontologies based on PR-OWL in an MTheory manner and it allows to save the built probabilistic ontology, namely the MEBN model, in PR-OWL format. The UnBBayes also provides a built-in MEBN reasoner.

Figure 4. PR-OWL simple model.

Figure 5. Mapping of PR-OWL random variables and OWL properties.

2.1.3. Context modeling in the underwater robot field

There is an upsurge in using ontologies to enable a formal representation for the robotic field [73]. The IEEE RAS Ontologies for Robotics and Automation Working Group [74] has been developing a standard ontology model [75] to represent the knowledge and reasoning in autonomous robots, such as air, ground, and underwater vehicles. However, this ontology focuses on presenting a very high-level service representation for vehicles themselves, such as sensors, resources, capabilities, platform, tasks, and mission. Context, which is useful to characterize the operational environment of vehicles, is not considered by this ontology. The ontology proposed by Insaurrealde *et al.* [76] also lacks the inclusion of environmental context, though it provides a good representation of planning and control systems for AUVs. The KnowRob [77] knowledge processing framework developed a set of ontologies to abstract robot actions, events, objects, environments, and the robot's hardware as well as inference procedures that operate on this common representation. The KnowRob puts its emphasis on improving the autonomy of individual vehicles instead of enabling the cooperation and coordination of multiple

vehicles. In addition, the household robots, rather than underwater robots, are the target modeling domain of the KnowRob ontologies. Similar to KnowRob, the ORO [78] system also leveraged a core robotics ontology to integrate data from diverse sources, such as sensors, domain knowledge, and human input. Nevertheless, the focus of the ORO ontologies is to help robots in interactions with humans. The RobotML [79] ontology, also known as PROTEUS ontology [80], was developed in the framework of the French research project PROTEUS [81]. The RobotML ontology aimed to enable scientific knowledge transfer between different robotic communities by formalizing the robots, their environments, robot parts, operations, mission, planning, their detailed behaviors, scenarios, *etc.* However, due to its complexity, the developed RobotML ontology cannot be directly used for exploitation as users must perform the semi-automatic transformation from the ontology to a UML representation. Besides, the RobotML is quite specific to their application, thus, it lacks generality to be reused in the underwater robot field. A reference ontology of collective behavior of autonomous agents and its extensions [82] was developed by Gorodetsky *et al.* This ontology provides model of the system and environment, model of interaction between the system objects and the environment, and model of the behavior of the system objects and the environment. Ontologies were also extensively adopted to provide semantic formalizations for many robotic applications, such as monitoring of the execution of robot plans [83], robots task planning [84], navigation planning [85], urban search and rescue missions with robots [86], space exploration using robots [87], and supervision of underwater environments for robots [88]. All these application specific ontologies for robots are conceived at a too specific and limited level to cover the overall modeling requirements from the cooperation of underwater vehicles. It is also worth noting that a common lack exists in all the aforementioned robotic ontologies, except the KnowRob framework [17]. They assume a deterministic world without considering context uncertainties, let alone properly model them.

2.1.4. Summary

Based on the overview of existing context and uncertainty modeling, ontology is chosen as the fundamental modeling method to provide a formal representation of information in the SWARMs project. Existing robotic ontologies have been reviewed in section 2.1.3. It is found that none of the existing robotic ontologies can provide a good representation of information in the context of the SWARMs project. Therefore, a new ontology for the representation of coordination and cooperation of underwater vehicles based on existing relevant robotic ontologies should be proposed. It should be general enough to cover necessary context from different domains, such as mission & planning, communication & networking, robotic vehicle, and environment recognition

& sensing. In the underwater robot field, context uncertainty comes from objective sources, such as imperfect instruments (sensors, cameras *etc.*) and unstable communication (data loss). Especially, the following uncertainty features, including ambiguity, vagueness, inaccuracy, incompleteness, and randomness, are inherent to context data obtained in the underwater environment. In the context of the SWARMS project, to address the context inaccuracy, incompleteness, and randomness, is considered as the primary target. Secondly, context ambiguity and vagueness are also going to be studied in order to seek potential solutions. The different context uncertainty features aforementioned should be properly modeled into ontologies for the sake of completeness. In addition, a comprehensive representation of context and its uncertainty can also be useful for further context reasoning. Probabilistic ontology is adopted to annotate uncertainty (focusing on incompleteness, inaccuracy, and randomness) based on the MEBN theory. Fuzzy ontology is also identified as useful to be used to address context ambiguity and vagueness in the context modeling of the SWARMS project. With the inclusion of uncertainties, the proposed SWARMS ontology can not only comprehensively represent the world of knowledge but also provide the ease of using a reasoning system.

2.2. Context reasoning

2.2.1. Review of existing context reasoning methods

To date, there are a set of context reasoning methods, such as rule-based reasoning, probabilistic reasoning, fuzzy logic, ontological reasoning. The detailed specifications for those context reasoning methods can be found in [1] [89]. It is noted that all the individual reasoning techniques have different weaknesses, explicitly described in [12] [89]. For instance, the ontological and rule-based reasoning could not handle uncertainty in their reasoning process. The importance of employing multiple reasoning techniques to mitigate individual drawbacks by using others' advantages is highlighted [90]. For example, Pilato *et al.* [91] applied ontological and BN reasoning to derive high-level context toward marine awareness. A combination of ontological and rule-based reasoning was adopted to intuit user activities [92]. Bobillo *et al.* [62] also introduced fuzzy logic into ontologies to provide a combined reasoning. The emerging probabilistic reasoning method, MEBN (already introduced in section 2.1.2), which is based on BN reasoning, shows good performance in reasoning over uncertainties in some domains, including procurement fraud detection [93], maritime awareness [94], knowledge-driven analysis for cultural heritage [95], and Robocup Soccer [96].

To conclude, the selection of context reasoning technique is subject to two factors: the performance and the supports from the modeling technique used. To design a specific application, the selection of the appropriate modeling and reasoning technique should be made carefully taking as many criteria and requirements as possible into account.

2.2.2. Summary

It is found that none of existing context reasoning methods are versatile and could individually solve all the reasoning requirements in underwater robots. There is a necessity for a shift toward a combination of different context reasoning methods. Since ontologies are the adopted solution to model information exchanged between vehicles in the SWARMS project, it is reasonable to apply ontological reasoning to check ontology consistency, concept subsumption *etc.* In addition, SWRL (Semantic Web Rule Language) [97] rules are proposed to be incorporated into ontologies and could mitigate the lack of ontologies in determining useful information based on rules. However, both of them lack the ability to reason under uncertainties as they assume a deterministic world. Since probabilistic ontology is adopted to annotate context uncertainty based on the MEBN theory. The Multi-Entity Bayesian Network (MEBN), as an extension of the standard Bayesian Network (BN), is chosen to reason about context uncertainty. It could represent and reason about problems that involve a varying number of uncertain entities. Such complex reasoning problems are common in the underwater robot field. For example, the investigated robot may have more than one robot around. The collision risk is affected by all possible nearby robots, thus resulting in a problem that involves a varying number of entities. The MEBN shows a lot of potential in reasoning about such complex problems in underwater robots. Therefore, in this thesis, the first proposal of using a hybrid context reasoning mechanism, including the ontological, rule-based, and MEBN reasoning, in underwater robots is presented. The incorporation of ontological, rule-based, and MEBN reasoning methods is envisaged to provide a better performance by compensating individual weaknesses using others' strengths. The proposed reasoning mechanism is more flexible and effective to provide different reasoning capabilities to satisfy different reasoning needs. It could provide a logic reasoning strategy to deal with certain context information while enabling a probabilistic inference on uncertain context information.

2.3.Context-aware framework in the underwater field

2.3.1. Existing context-aware frameworks

This section reviews existing context-aware frameworks. Since there are not many context-aware framework proposals in the underwater robot field, context-aware frameworks defined in other fields, which can inspire the design of underwater robot context-aware framework, are also included.

TRIDENT proposal

A multisensory control architecture, including a knowledge-based approach, to guarantee the suitable manipulation actions for enabling a multipurpose intervention system, is proposed in the TRIDENT project [19]. To this end, a cooperative team formed with an Autonomous Surface Craft (ASC) and an Intervention Autonomous Underwater Vehicle (I-AUV) is used to complete a series of sequential activities, including Survey (launching, survey, and recovery) and Intervention (launching, approaching, intervention, and recovery). The TRIDENT project is claimed to go beyond present-day methods typically based on manned and/or purposed-built systems by advocating the usage of this proposed architecture. It could offer intervention tasks for diverse potential applications like underwater archaeology, oceanography, and offshore industries. TRIDENT project highlights its study on intelligent control architecture to provide an embedded semantic and context-aware framework and high-level reasoning capability required to enable a high degree of autonomy and onboard mission decision making.

The semantic world model framework proposed in [88] is conceived to provide a capable and holistic system bent on integrating a set of autonomous underwater robots in which semantic interoperability is achieved among various information sources. This semantic framework embedded in robot architecture is believed to enhance interoperability, independence of operation, mission flexibility, robustness, and autonomy. The whole architecture can be seen in Figure 6.

Figure 6. Semantic and context-aware framework for autonomous underwater vehicles.

The decision-making process heading for the realization of context awareness within this framework is based on OODA (Observe, Oriented, Decision, and Act) [98] as Figure 7 shows. The objective of context awareness is to make the vehicle to autonomously understand the big picture. This picture is outlined by the combination of experience achieved from previous missions (orientation) and the information obtained from the sensors while on mission (observation).

Figure 7. OODA loop for processing data.

A hierarchical ontology model (depicted in Figure 8) plays a crucial role in enabling context awareness by laying the knowledge representation foundation and guiding the decision-making process.

Figure 8. Context awareness concepts.

Different ontologies are proposed at both core-oriented and application-specific levels. For example, a set of application ontology, which provides an underlying formal model for tools integrating source data and performing a variety of extended functions, includes status monitor application ontology and mission planning ontology. Core ontology includes several concepts like Platform, Payload, Module, Sensor, and Driver. Ontologies are utilized to represent the raw data obtained from robots and finally they are available for high-level decision-making agents. However, environment related context information is not included in the ontologies. In addition, context uncertainty is not considered in the TRIDENT framework, neither in the modeling nor in the reasoning phase.

Context-aware middleware for pervasive elderly homecare (CAMPH)

CAMPH [99] is a framework to glue hardware infrastructure with various context-aware applications, especially with the emphasis on the pervasive homecare area. Generally, this framework offers several key-enabling system-level services, including context data acquisition, context storage, context reasoning, context query processing, service organization, and discovery. The overall architecture, as can be seen in Figure 9, consists of four logical layers: Physical Space Layer, Context Data Management Layer, Service Management Layer, and Application Layer.

- *Physical Space Layer:* Each Physical Space (PS) may contain physical entities such as sensors, actuators, and computing devices. At this layer, data is modeled as a pair of attribute-value. PSs with high relevance, such as having similar attributes, are sorted into the same class named context domain or context space.

- *Context Data Management Layer:* Main components allocated in this layer are Context data/events database, Query processing, Context space management, and Context reasoning. A hierarchical reasoning scheme, in which low-level reasoning performs on single PS data while high-level reasoning makes inference from context domain, is applied. Besides, SQL-based context query interface is available to acquire context data or subscribe to event notifications.
- *Service Management Layer:* Context data is fully utilized to enable context-aware service organization and discovery.
- *Application Layer:* Different homecare applications can invoke and orchestrate context-aware services or make requests for context data directly in this layer.

An example of a personalized homecare application mashed from web services running this middleware was developed to demonstrate the working principles of CAMPH. However, this prototype is still far from usable. The usability of CAMPH should be examined in a larger scale field trial. Besides, context data exchangeability/interoperability among different context domain is a concern since context data is structured in the key-value model. How to manage the massive amount of context data from various spaces is still an unsolved issue. Although it is declared that a hierarchical and comprehensive reasoning scheme will be deployed, explanation about detailed procedures is still missing. Security and privacy are not considered during the entire conception/design.

Figure 9. The architecture of CAMPH.

ACoMS+

ACoMS+ [100], as an enhancement of ACoMS [11], offers a solution for resource-efficient and context-aware management of sensing infrastructure. As can be seen in Figure 10, the core of ACoMS+ is composed of a Context Source Manager (models raw context information and performs actions on low-level communication), an Application Context Subscription Manager (allows applications to subscribe interested context by specifying quality of information and service), and a Reconfiguration Manager (reconfigures sensing devices to offer fault-tolerant

provisioning of information). Context Modeling Language (CML) is selected as the modeling method which leverages the graphical notations to represent context information.

To make the reconfiguration process more elaborate, a mining algorithm called HiCoRE [101] is incorporated into ACoMS+. While dealing with different applications, decisions can be made wisely based on those correlated rules of context which are mined by HiCoRE. ACoMS+ adopts the HiCoRE algorithm to fully utilize operational objectives and discover correlations among sensing infrastructure. However, HiCoRE is able to mine and rank correlations on the premise that every sensing entity can only perform a single sensing aspect (such as temperature, humidity). Future efforts can be focused to make the mining algorithm more comprehensive by taking into account complex sensing devices (e.g., camera, tablet) with multiple sensing capabilities at the same time. Besides, the presented design has just been partially implemented and tested. Continuous research is still needed to improve this proposal.

Figure 10. ACoMS+ architecture.

FIWARE

The FIWARE [102] FP7 project has an ambitious intention to strengthen the competitiveness of the EU economy by presenting a cutting-edge infrastructure in which creation and delivery of services, high Quality of Service (QoS), and security are enabled. This platform is conceived to be considerably generic and could adaptively fit into various usage areas, e.g., safety, logistics, environment, energy, traffic and mobility, and agriculture. This platform is built based on a public cloud with a rich library of modules offering various added-value functions (referred as services). These modules, regarded as Generic Enablers (GEs), fulfill all the capabilities of different chapters of this architecture, such as service delivery, cloud hosting, Internet of Things, support services, developer tools, and interface to the network and devices.

Among all, it is worth stressing that enablers dedicated to managing context data from heterogeneous resources are playing an important role in the whole platform. These aforementioned enablers could be grouped into two categories: the Semantic Virtualization Enablers and the Cognitive Enablers.

Three main concepts, which are Actors, Resources, and Applications, are virtualized to lay the basic foundation of the semantics of this cognitive middleware. The Semantic Virtualization Enablers are responsible for abstracting the heterogeneous Actors, Resources, and Applications by means of attaching homogeneous, context aware and semantic aggregated metadata. On the basis of semantically abstracted metadata available through well-defined Restful Application Programming Interfaces (APIs), Cognitive Enablers are capable of making decisions regarding the best solution of exploiting the available resources to efficiently satisfy application requirements and needs. To meet a specific application's needs, different Cognitive Enablers can be dynamically orchestrated. In the real deployment, all the functionalities of enablers can be distributed into different physical network entities.

The design of the Semantic Virtualization Enablers and Cognitive Enablers allows Applications to transparently, efficiently, and flexibly employ available Resources and become customer-tailored. With the collaboration of other Enablers, such as Security Enablers, Cloud Hosting Enablers, and IoT Enablers, the proposed FIWARE is upgraded to be a full-fledged middleware with context awareness, interoperability, cloud hosting, big data analytics, and service delivery and composition. What is more appealing for developers is that, so far, all the Enablers developed in the FIWARE project are available as open-source implementations associated with detailed user manuals. However, it still will not be an easy task to move the pilot tests to real commercial usages. More potential constraints introduced by harsh environments such as underwater and off-shore should be taken into account when adapting FIWARE to actual operations.

Context awareness for Internet of Things (CA4IOT)

A sensing-as-a-service framework presented in [103] is called Context Awareness for Internet of Things (CA4IOT). It appears that this middleware is conceived merely to solve a single issue of how to select the most suitable sensors according to the tasks/problems at hand rather than providing a complete solution for managing context data. An overview of CA4IOT architecture is displayed in Figure 11.

Figure 11. CA4IOT architecture.

Four major layers that form the framework are listed as follows:

- *Sensor Data Acquisition Layer (SDAL)*. The main components located in this layer are sensor wrappers, wrapper repository, wrapper generator, sensor device definition (SDD) local repository, and SDD cloud repository. This layer is responsible for acquiring a variety of context data.
- *Context and Semantic Discovery Layer (CSDL)*. This layer is in charge of discovering context and semantic. Relevant components are context and semantic discoverers, context and semantic discoverer generator, and context and semantic discoverers repository.
- *Context Processing and Reasoning Layer (CPRL)*. A collection of important functions is distributed in this layer like processing data, reasoning high-level context, fusing context, and knowledge generating and storing.
- *Data, Semantics, and Context Dissemination Layer (DSCDL)*. Users can make requests via multi-model interfaces. Local repository can interact with repositories which reside in the cloud or open linked data to provide better answers for those queries by means of big data analytics.

In the CSDL and CPRL layers, context data related to sensors are represented in XML. In addition, users submit their requests for querying context using XML data format. In general, this framework is quite complex and mature with detailed explanations for the different functional component. Many technical factors, such as context abstraction, context process, and context dissemination, are carefully considered while context uncertainty is neglected. Compared with other framework solutions, this framework can act as a standalone architecture or an auxiliary

technique to be integrated with other framework solutions to fulfill the demands from different paradigms. However, only a simple use case is employed to show the specific procedures of developing the proposed middleware and implementation of this middleware is still missing.

Context-Aware Middleware for Pervasive and Ubiquitous Service (CAMPUS)

CAMPUS [104], short for Context-Aware Middleware for Pervasive and Ubiquitous Service, is proposed to automate context-aware adaptation decisions with the influence of three key technologies: compositional adaptation, ontology, and description logic/first-order logic reasoning. It has taken an enormous step to advocate automated run-time adaptation decisions instead of depending on predefined adaptation policies that only take limited contextual changes potentially operating in a dynamic situation.

As can be seen in Figure 12, CAMPUS, as a typical layered architecture, consists of three tiers: the programming layer, the knowledge layer, and the decision layer.

- *The programming layer.* It is responsible for constructing and reconfiguring context-aware applications by adopting the instructions from the decision layer.
- *The knowledge layer.* Three ontologies including Context Model, Tasklet Model, and Service Model are proposed to represent the semantics of knowledge which is necessarily required by CAMPUS to make adaptation decisions. The knowledge could be the requirements desired by target service, the properties of the available tasklets, the context requirements imposed by tasklets, and the properties of run-time context.
- *The decision layer.* Decision maker uses a multi-stage normative decision model, which includes preprocessing, screening and choice, to choose the best tasklet alternatives for a given task. The automated adaptation decisions will be forwarded to the programming layer.

CAMPUS provides an effective middleware solution for integrating context awareness to application development. CAMPUS could automatically derive context-aware adaptation decisions at run time by means of semantic-enhanced decision making. Nonetheless, security has not been mentioned in this proposal. Context uncertainty is not considered, either. Collaborative decision making among multiple CAMPUS middleware instances can be a future extension.

Figure 12. The architecture of CAMPUS.

Context-Aware Services Framework (CASF)

A context-aware services framework proposal, here abbreviated as CASF, aiming at providing a variety of context-aware services was presented in [105]. The authors Juyoung *et al.* realized that many context-aware frameworks lack service discovery and composition capability. Consequently, they came up with this new architecture aiming at tackling this gap. Basically, this framework is built based on semantic web services as they are well-known for supporting automatic service discovery and integration. To achieve the integration of services, which also refers to as selection and combination of context information, this proposal separates context-aware services with context-aware information. The core of this architecture, named as context mediation framework, is shown in Figure 13.

As can be seen in Figure 13, the context mediation framework consists of three different tiers: physical sensor layer, public context layer and context service layer.

- *Physical sensor layer.* It can only recognize sensor data. Physical sensors are the only context information source.
- *Public context layer.* Two types of context providers in terms of complexity of context information process are located in this layer. A basic context provider only processes sensor data from physical sensors while a combined context provider can make use of information from both sensors and other context providers. All context information generated in this layer is served based on web services so that openness and

interoperability are achieved. By using a proposed context ontology and OWL-S [106], context providers are able to be constructed in web services.

- *Context service layer.* Context information is consumed in this layer so that context-aware services can be generated and provided to users.

Figure 13. The architecture of the CASF.

The major novelty of this proposal is the adoption of the concept of semantic web services. By publishing context information based on semantic web services, it is also feasible to achieve automatic discovery and integration for context information. However, many follow-up studies should be conducted to make the proposal complete. Firstly, more detailed protocols and ontologies should be specified to translate context information to web services. e.g., SOAP [107]-based messaging protocol could be adopted to connect the communication between public context layer and physical sensor layer with more details explained. Besides, the CASF is still under development. This architecture lacks prototyping test by building various advanced context-aware services with regard to real environments.

Semantic Web-Based Context Management (SeCoMan)

SeCoMan [108], as the abbreviation of Semantic Web-based Context Management, is intended to provide a privacy-preserving solution for developing context-aware smart applications. In SeCoMan, ontology is employed to model the description of entities, reason over data to obtain useful knowledge, and define context-aware policies. The whole architecture of SeCoMan is shown in Figure 14.

All the modules making up the SeCoMan are placed in a layered structure including Application, Context Management, and Plug-in (from top to bottom).

- *Application.* Different applications reside on top of SeCoMan in order to offer desired services for users.
- *Context Management.* As the core of the SeCoMan framework, it provides context-aware supports for applications. Three kinds of actors with different rights to interact with SeCoMan are defined including Framework Administrator, Application Administrator, and Users. A set of predefined queries is allowed for applications to get information about the indoor location of users and objects. Semantic rules are used to specify policies regarding restricted access to location information so that privacy is guaranteed.
- *Plug-in.* It provides SeCoMan with context information which is especially focused on locations. In other words, the plug-in layer acts as an independent context source.

Figure 14. SeCoMan, as described in [108].

In fact, the so-called context-aware solution, offered by SeCoMan, is limited to get aware of locations. Therefore, privacy protection is fulfilled in a location-limited level which enables users to share their location with the right users, at the right granularity, at the right place, and at the right time. SeCoMan, especially the context management layer, is planned to be integrated into the cloud architecture in future work. In this way, this middleware could take advantage of the features of cloud computing to achieve extra capabilities, such as elasticity, monitoring, load

balancing, and address security issues. Besides, the current privacy scheme will be augmented by introducing anonymity and hashing policies to hide and disguise the identity of a user. The exploration of outdoor usage could be the next step to improve the generality of this framework.

Cloud-oriented Context-Aware Middleware in Ambient Assisted Living (CoCaMAAL)

Forkan *et al.* [109] presented a novel Cloud-oriented Context-Aware Middleware in Ambient Assisted Living (AAL) which is abbreviated as CoCaMAAL. The motivation behind is that biomedical sensors which are widely used in AAL lack the processing power to perform key monitoring and data-aggregation tasks. Therefore, cloud computing is adopted to address computing needs. In particular, this proposal is believed to serve as a scalable and context-aware framework which can ease the flow between data collection and data processing in AAL scenarios. Basically, CoCaMAAL is built on the basis of Service-Oriented Architecture (SOA) which performs context modeling for raw data, context data management and adaption, context-aware service mapping, service distribution, and service discovery. CoCaMAAL comprises five main cloud-oriented components: AAL systems, context aggregator and providers (CAP) cloud, service providers cloud, context-aware middleware (CaM) cloud, and context data visualization cloud.

- *AAL systems.* This component, as the hardware architecture, includes different BSN (Body Sensor Network) foundations and monitoring systems for meeting different target user requirements.
- *Context aggregator and providers (CAP).* Raw data from AAL systems is converted and abstracted to high-level context by CAP. More specifically, context providers categorize sensor data into context based on pre-designed ontology. For instance, Person, Place, Environment, and Device are the main entities defined in the context ontology and they are associated with different attributes. Afterward, different context is integrated by context aggregator to provide complete information. In addition, reasoning mechanisms are applied to infer more useful information.
- *Service providers.* They are the producers of context-aware services, such as applications.
- *Context-aware middleware (CaM).* By utilizing existing knowledge and incoming context, CoM is able to identify assistive services for the given context and trigger associated actions. CaM is the core component of CoCaMAAL with multiple key functions, such as context management, context storing, context retrieval, context manipulation, service mapping, self-adaptation, service discovery, and security service.
- *Context data visualization.* Proper interfaces (e.g., GUI) are available for users to visualize context data.

A prototype based on CoCaMAAL was developed in Java. The implementation examines the performance of the proposed architecture, such as the influence of increasing context and service load on service response time. The results prove that CoCaMAAL is efficient at collecting, abstracting and using context from AAL environments to provide context-aware services. The major novelty is the adoption of cloud computing which provides powerful computing capabilities to process context. However, several concerning issues cannot be ignored, *e.g.*, conflicts in context are not considered, reliability analysis is not accomplished, and context uncertainty is omitted. Although Forkan *et al.* [109] stated that the context aware role-based access control and privacy-preserving context service protocol can be adopted to ensure privacy in this middleware, those two mentioned approaches are not included in the test.

Big Data for Context-Aware Monitoring (BDCaM)

Motivated by CoCaMAAL, a novel context-aware framework architecture, named Big Data for Context-aware Monitoring (BDCaM) [110], is proposed. As an extension of CoCaMAAL, BDCaM addresses additional concern: personalized knowledge discovery. The underlying approach of discovering personalized knowledge is to derive/learn patient-specific anomalies from amounts of data. The adoption of a novel learning process in BDCaM is an important step forward. A 2-step learning methodology is newly proposed in [110] to derive more useful information for context-aware decision making. The specific procedure of this learning approach is as follows: firstly, correlations between context attributes and threshold values are identified. Possible association rules which are patient-tailored will be generated by applying the MapReduce Apriori algorithm [111]. Finally, supervised learning is performed over context data based on those rules generated in the first step. Like CoCaMAAL, BDCaM is split into several distributed and cloud-based components which are Ambient Assisted Living (AAL) Systems, Personal Cloud Servers (PCS), Data Collector and Forwarder (DCF), Context Aggregator (CA), Context Providers (CP), Context Management System (CMS), Service Providers (SP), and Remote Monitoring Systems (RMS). A use case related to health monitoring is implemented on this middleware and implementation results have proven the applicability of this framework and the efficiency of detecting patient's anomalies. However, context uncertainty is not modeled and considered in the reasoning process. Besides, exploring the possibility of generalizing this framework to suit more domains (not limited to AAL) could be a further improvement.

FlexRFID

A recently published proposal called FlexRFID [112] aims to provide a policy-based framework solution for facilitating the development of context-aware applications and integrating

heterogeneous devices. Ponder is adopted as the policy specification language in this middleware. The FlexRFID middleware, as can be seen in Figure 15, is a multi-layered framework consisting of Device Abstraction Layer (DAL which abstracts the interactive operations among the physical network devices), Business Event and Data Processing Layer (BEDPL which provides context data management like aggregation, transformation, and dissemination), Business Rule Layer (BRL that manages policy-related operations), and Application Abstraction Layer (AAL that enables communications among applications and the FlexRFID). FlexRFID is claimed to provide all data processing capabilities like filtering, grouping, dissemination, and duplicate removal. In addition, it is an enabling solution to support simultaneous communication among different applications which are built in this framework. Notably, FlexRFID differs from other context-aware frameworks in the capability of policy enforcement. A plethora of benefits can be achieved by defining different types of policies such as ensuring privacy, constraining access control, and offering customized services. Two abstract types of policies are stated in FlexRFID: System Policies (manage the operations done by the middleware) and Application Policies (define the way users want the FlexRFID services to be delivered).

Figure 15. The FlexRFID framework.

The authors in [112] focus mostly on implementation details and performance evaluations of FlexRFID. Experimental results obtained from two real scenarios (healthcare and book management) show that the response time will become longer as the volume of policies increases. Also, for this reason, some specifications for the concrete techniques used in the middleware are missing in this paper. For example, a data formalization method is not included. The current version of the FlexRFID middleware only offers basic security mechanisms by means of specifying access control policies. More advanced security measures should be taken, *e.g.*, application authentication and security at the level of tags and sensor nodes. Further improvement could be integrating FlexRFID in the cloud so as to enable applications to flexibly use cloud-based services and adapt those services with regard to specific application policies and context considerations.

2.3.2. Summary

After reviewing a set of existing context-aware frameworks, it can be found that currently introducing context awareness in the underwater robot field, especially in cooperating underwater robots, is under-researched compared with that in other fields, such as smart home and AAL. Although existing work cannot meet the requirements in the underwater robot field and be directly used in the SWARMs project, all of them can provide valuable inspirations for the brand-new design of a context-aware framework for the cooperation of underwater robots. It is important to notify that the review on existing context-aware frameworks has revealed several common issues existing in the current context-aware framework. Specifically, they are listed and described as follows:

- **Low semantic capability.** Context needs to be modeled in a formalized way. Different context modeling method differs from each other in terms of formality and interoperability. In this case, it is assessed the degree of semantics that is present in the proposal. Semantics are usually of major importance in order to extract information of the system and be able to infer and learn about the context where the deployment is done. What is more, for underwater robotics it is even more important, as AUVs may be put in the situation of having to take decisions by themselves, so a significant degree of semantic capabilities must be provided for them. If ontology is the chosen modeling method, the use of existing ontologies is considered as a parameter to assess if the context model is reusable and interoperable. It is common that some existing context-aware frameworks do not reuse existing ontologies.

- **Lack of multimodal context reasoning.** No single reasoning model can accommodate the demands of real-world applications. Each context reasoning method has its strengths and limitations. Most of existing context-aware frameworks only adopt one context reasoning algorithm resulting in a limited reasoning capability. An ideal context-aware framework should incorporate multiple reasoning methods together to complement each other's strengths and mitigate their weakness.
- **Lack of the capability of dealing with uncertainty.** Context uncertainty is the norm than the exception in real-world applications in the underwater environment [113]. The context-aware framework should be capable of dealing with different kinds of context uncertainty, such as context ambiguity, context vagueness, and context ambiguity. More specifically, it should provide proper modeling of context uncertainty and support effective reasoning on the modeled context uncertainty. None of the existing context-aware frameworks provide a principled representation of context uncertainty and provide support for uncertainty reasoning. In the underwater robot field, it is necessary to encase the framework with this capability.
- **Unfollow an ontology development methodology.** Most of the existing context-aware frameworks use ontologies to represent heterogeneous context. However, they do not follow any ontology development methodology in their design. The way ontology is designed could be considered as a parameter to assess the context-aware framework. A development methodology can provide a formal guidance for the design process and guarantee the quality of ontology in terms of completeness. A well-organized schedule of activities proposed by ontology development methodologies can provide methodological supports for ontology engineers. To date, there are a set of existing ontology development methodologies, such as NeON [114] and Methontology [115]. They fare better or worse and in nature are subjective. However, the adoption of an ontology development methodology can somewhat guarantee the ontology design with a better quality.

All the aforementioned open issues should be carefully resolved when designing a new context-aware framework for underwater robots. In addition, the following technical features should also be considered.

- **Architecture Style** defines the way the context-aware framework is constructed. It is the most elementary factor to be initially considered. Four classical architectural fashions can come in handy to organize and arrange the inner composition of middleware, including stand-alone, centralized, layered, and distributed. The distributed manner is widely applied and it enables the design in a more flexible manner. Different components hold distributed

responsibilities and they are independent of each other. The functionalities of the context-aware framework should be divided into layers or components in a meaning manner. Each component should perform a limited amount of the task and should be able to perform independently up to a large extent.

- **The storage of context** is highly demanded as historical context is still meaningful for further use, such as context reasoning. An appropriate storage container should be carefully chosen to store context and its rich semantics by taking into account its data volume.
- In order to correctly implement context awareness, **context extraction and processing** are time and resource consuming. There is always a trade-off between the amount and quality of the gathered information and the actual gain achieved from a more accurate context. Determining this balance is one of the challenges for underwater robot context-aware frameworks. Policy constraints can be applied at early stages to reduce the wasted resources in context extraction.

3. The context-aware framework for the cooperation of underwater robots

Based on the analyses on existing context-aware frameworks, a new context-aware framework is proposed in this thesis in order to provide a complete and well-defined management for context data that are exchanged between vehicles, such as information about the underwater environment, mission and planning, communications, and context from external data sources (e.g., marine experts and third-party information providers). In general, the context-aware framework should be able to understand available context sources, their data structure, and automatically build internal data models to facilitate them. Further, it should provide functionalities to transform the retrieved raw context information into appropriate context representation models correctly with a minimum amount of human intervention. Specifically, the objective of the context-aware framework is to abstract heterogeneous context in a unified format, enable data and capabilities sharing between vehicles, encase data with semantics, reason about context for high-level information, disseminate relevant data to entities, and provide a homogeneous application development interface. Thus, the SWARMs general architecture can be augmented with the context awareness feature provided by the context-aware framework. Services provided by the context-aware framework can be exploited in diverse ways, e.g., vehicles can have a better understanding of the operational environment and operators can optimize the mission plan by making use of high-level information reasoned by the framework.

The context-aware framework is conceived to be modular and distributed, which could imply a better potential to be employed in robot coordination and cooperation versus a centralized framework one. Specifically, as can be seen in Figure 16, the context-aware framework consists of six logic components, namely, **Ontology Model**, **Data Processor**, **Semantic Mapper**, **Rules Creator**, **Context Reasoner**, and **Semantic Query**. The functional capabilities of each component are described as follows:

- **Data Processor.** The extreme underwater conditions impose more challenges in the data acquisition phase. Thus, context data obtained from the environment could be uncertain. A preliminary treatment for uncertain data is needed and this treatment is provided by the data processor component. This component can pre-process the data obtained by sensors, vehicles or other sensing instruments. Statistics, such as probability distribution, can be learned using machine learning algorithms in this component. Operations, such as validating values, checking inconsistencies, calculating uncertainty degrees, removing outliers, and filling in missing values, can be executed in this component.
- **Ontology Model.** This component, acting as a semantic repository, keeps the SWARMs ontology and stores all data obtained as instances in the ontology model. Ontology is

adopted to serve as a common information model to represent information and enable sharing, reuse, and integration of data between vehicles. The information model is structured in a hierarchical manner. The networked information model consists of three levels of ontologies: core ontology (acting as an upper-level ontology to glue all domain specific ontologies), domain-specific ontology (providing information models for different domains, e.g., mission planning, vehicles, environment, and communications), and application-specific ontology (describing information with a focus on particular applications, e.g., oil spill detection, plume tracking, and berm construction). Therefore, data, including but not limited to contextual measurements, vehicle-related data, data from external sources (Global Positioning System, oceanic weather forecast, etc.), and marine experts' knowledge, can be abstracted and formalized in an ontological format. All data can be displayed with a homogeneous view associated with the semantic content. The ontology model could annotate context uncertainties based on MEBN theories. Specifically, probability information about uncertain context can be modeled using ontology constructs defined in the PR-OWL ontology. With PR-OWL classes and properties, ontology engineers could fully specify an MEBN model while maintaining compatibility with the OWL ontology language. The SWARMs ontology will be detailed in chapter 4.

- **Semantic Mapper.** This component plays a vital role in facilitating the transparent sharing of information. As data might be formatted in different manners pertaining to different data sources, this component aims to parse and formalize them in an ontology-compliant format. Translations from different standards, such as XML, JSON, or binary files, to ontological formations, such as RDF or OWL, can be enabled in this component. For differently formatted data that do not comply with the SWARMs ontology, corresponding mapping files should be predefined in order to parse and map them into the common information model.
- **Rules Creator.** Operators or marine experts are able to define rules based on their knowledge before or during missions through this component. A set of user-defined rules can be translated into the SWRL format and inserted into the ontology model. Rules can be diverse, including restrictions or definitions for entities, regulations for evaluating data values, or specifications for relationships between entities. The rule set is the input for the context reasoner to make the rule-based inferences.
- **Context Reasoner.** The essential capability of this component is to derive new knowledge from available context information stored in the ontology model. Basically, it will consult

information stored in the ontology model and also take experiences and knowledge from marine experts into account. A hybrid reasoning mechanism, including the ontological, rule-based, and MEBN reasoning, is intended to be employed in this context reasoner. Specifically, the ontological reasoning enables several kinds of operations, including concept satisfiability, consistency check, class subsumption, and logic inference. The rule-based reasoning could augment the ontological reasoning in terms of logicity and human readability. The MEBN reasoning is dedicated to reasoning under uncertainties. The specific hybrid context reasoning mechanism for this component will be detailed in chapter 5.

- **Semantic Query.** This component deals with any semantic query made by operators. It receives queries and consults the SWARMs ontology to get answers. This is one of the context dissemination strategies defined in the SWARMs project. Apart from this, the SWARMs project also follows a subscribe/publish paradigm to distribute context information.

The context-aware framework provides architectural support for different context treatments. Within the framework, the ontology model and the context reasoner are significant in realizing context awareness. By cooperating them, it is able to provide necessary services to meet all the modeling and reasoning requirements in the SWARMs project. In particular, it can handle the intricacies inherent to the context uncertainty and its reasoning.

The context-aware framework is designed as part of the SWARMs middleware architecture. It is of nature that the context-aware framework closely interacts with other middleware components in a dual way, either provides functionalities to middleware components or uses services provided by other middleware components. Especially, the data access manager defined in the SWARMs middleware architecture interacts with the context-aware framework most. It provides interfaces able to insert/retrieve/update information in the ontology model component. Specifically, it provides different methods to update the deterministic knowledge base and the probabilistic knowledge base in the ontology model, respectively.

Figure 16. Overview of the proposed context-aware framework.

4. Context modeling for the cooperation of underwater robots

The context modeling in the cooperation of underwater robots will be presented in this chapter. Specifically, three major contributions that this thesis provides regarding the context modeling topic will be introduced. Chapter 4.1 will review existing ontology development methodologies, propose a new fuzzy ontology development methodology, and review a probabilistic modeling methodology. By following the proposed fuzzy ontology development methodology and the probabilistic modeling methodology, an ontology, named the SWARMS ontology, is proposed and it will be specified in chapter 4.2. In addition, two examples are presented to extend the capability of the proposed SWARMS ontology in modeling fuzzy and probabilistic knowledge, respectively. Focusing on ontologies which have embedded mathematic relationships, the application of a Stochastic Reduced Order Model (SROM) to quantify uncertainties propagated in those mathematic relationships is firstly proposed in this thesis and it is introduced in chapter 4.3.

4.1. Ontology development methodologies

4.1.1. Existing ontology development methodologies

In this section, the state of the art in ontology development methodologies is presented. Specifically, a summary of the most well-known methodologies for building crisp ontologies is provided. In addition, existing fuzzy ontology development methodologies presented in current research are also reviewed.

It is widely accepted that there is no single "correct" way or methodology for developing ontologies [116]. Aiming to provide good guidelines for crisp ontology constructions, various ontology development methodologies have been presented. An ontology development methodology provides a formalization for scheduling activities or tasks that should be followed and performed during the design process. Workflows proposed by different methodologies might fare better or worse regarding efficiency, ease of use, comprehensiveness, and rationality. A well-organized schedule of activities proposed by ontology development methodologies can provide methodological supports for ontology engineers. The most well-known ontology methodologies proposed in the current literature are METHONTOLOGY, NeOn, DILIGENT [117], On-To-Knowledge [118], HCOME [119], and DOGMA [120]. In addition, Noy *et al.* [116] presented a very descriptive yet simple guide to creating crisp ontologies. A set of survey papers, such as [121] [122] [123], are also available providing good references to existing ontology development methodologies and their features. To conclude, a considerable amount of methodologies can

come in handy for developing crisp ontologies. However, these methodologies dedicated to crisp ontologies cannot be directly applied to construct fuzzy ontologies due to major differences between fuzzy ontologies and crisp ones. In order to develop fuzzy ontologies, additional procedures, such as including fuzzy logic to approximate vagueness and conceptualizing the fuzzified vagueness, should be considered in the development process.

Current research on fuzzy ontologies mainly focuses on dealing with conceptual formalisms. In other words, how to represent fuzzy ontologies in a formalized language is the most active work. How to develop fuzzy ontologies in a standard and effective way is under-researched. The IKARUS-Onto [124] methodology is a methodology for fuzzy ontology development. It focuses on the provision of a methodological guideline for the conversion from crisp ontologies into fuzzy ones. It consists of five formal steps, including acquiring crisp ontology, establishing the need for fuzziness, defining fuzzy ontology elements, formalizing fuzzy elements, and validating fuzzy ontology. The IKARUS-Onto methodology represents a comprehensive guidance for fuzzifying crisp ontologies. Thus, it is suitable to be used to develop fuzzy ontologies in domains with the existence of crisp ontologies. Similarly, the Fuzzy Ontomethodology [125] also emphasizes on formalizing the activities for developing fuzzy extensions based on available crisp ontologies. The Fuzzy Ontomethodology consists of three steps, including conceptualization, ontologisation, and operationalization. Processes grouped in each step are too ambiguous to be understood and used in practice. In addition, the Fuzzy Ontomethodology is devoted to providing guidelines for building ontologies for semantic web search. Reusing fuzzy elements (e.g., fuzzy concepts, fuzzy sets, fuzzy relationships, or fuzzy data types) that have been defined in existing fuzzy ontologies can enhance the interoperability and shareability in the ontology community as well as guaranteeing less workload. Nevertheless, neither of existing fuzzy ontology methodologies does consider the inclusion of an important step, which is reusing existing fuzzy ontology elements, in the development process. While attempting to model knowledge in domains where no existing crisp ontologies are available, the development of fuzzy ontologies should be guided in a formal way. Since existing fuzzy ontology methodologies rely on the existence of crisp ontologies, it is apparent that a methodological approach for developing fuzzy ontologies from scratch is still a lack in current literature.

Fuzzy ontologies should be built following a methodological guideline in order to better model imprecise and vague information. To this end, chapter 4.1.2 will present a fuzzy ontology development methodology which could provide well-defined engineering principles to improve the development and building of fuzzy ontologies from scratch. This proposed method could enable

good treatments and utilizations of vague or imprecise knowledge in terms of generality, accuracy, reusability, efficiency, and shareability.

Regarding the probabilistic modeling in the ontology based on the MEBN theory, Carvalho *et al.* [126] proposed a methodology which specifies the specific uncertainty modeling process for probabilistic ontology. This methodology will be introduced in chapter 4.1.3.

4.1.2. A fuzzy ontology development methodology (FODM)

The aim of the proposed FODM is to provide a formal abstraction of activities that need to be done throughout the development process. The proposed methodology is dedicated to presenting the first methodological approach to building fuzzy ontologies from scratch, rather than converting existing crisp ontologies into fuzzy ones. The whole workflow of the proposed FODM can be viewed in Figure 17. In general, all the activities or tasks are grouped into eleven phases to form the entire lifecycle of building a fuzzy ontology. Each phase and its associated purposes and activities are elaborated in the following subsections.

Figure 17. Overview of the fuzzy ontology development methodology.

Phase 1: Ontology purpose and scope

As defined in the majority of crisp ontology development methodologies, such as Methontology, the primary task is to clarify the motivation of building a fuzzy ontology. In other words, the purpose and scope of modeling information using fuzzy ontology should be clearly defined. Basic questions should be raised and explicitly answered in order to make the purpose and scope of ontology clear. For example, a set of questions could be 1) What is the domain or scope of information that needs to be modeled? 2) Is ontology the best modeling technique over

other solutions, such as text, key value, and Unified Modeling Languages (UML)? 3) What is the type (including domain-specific, generic or core, application specific, and representational ontologies) of ontology depending on the determination of domain or scope? 4) Who will be involved in the development of ontology and what roles they are going to play? 5) How to ensure a tight collaboration between different participants so as to guarantee a successful development of ontology? Once questions are accurately addressed, the purpose and scope of ontology could be established. Though answers to those questions might slightly change during the development process, the general purpose and scope could retain at given moments. Until now, it is clear that an ontology is going to develop in order to model information within a specific domain or scope.

Phase 2: Identify the need of fuzziness

With using fuzzy ontologies to manage vagueness and impreciseness born in mind, the second phase aims to identify whether fuzziness should be introduced into the ontology design. The ultimate goal of this step is to determine what type of ontology is going to build: either crisp ontology or fuzzy ontology. In this step, both ontology engineers and domain experts should participate and cooperate with each other to establish the need of fuzziness. The reason behind the involvement of domain experts is because domain experts could provide specialized knowledge to analyze if fuzziness is needed. To obtain a proper answer, a set of activities should be conducted. Firstly, a deeper identification on the domain or scope of the ontology should be done. A first check on the information that is going to be modeled can enrich the understanding of the necessity of fuzziness. After the check, information that is vague present in the domain or scenario could be found out. Secondly, domain experts will justify whether fuzziness will be taken into account in the ontology design. Before the emergence of the fuzzy ontology technique, crisp ontology is widely used in a diversity of domains where actually vague information exists. However, all information in those domains is assumed to be accurate and uncertainty inherent to information is neglected. Now with the fuzzy ontology technique, it is feasible to deal with vagueness that crisp ontologies could not. Nevertheless, the need of fuzziness should be decided by domain experts because of the balance between the degree of vagueness and complexity of building fuzzy ontologies. In other words, to what extent the planned ontology is going to represent the information should be justified. Thirdly, fuzziness might exist in different ontology elements according to the definition of fuzzy ontologies. Different types of fuzziness should also be identified, such as indetermination of individuals in instantiating concepts (namely, fuzzy concepts) and blurry relations in pairs of individuals (namely, fuzzy relations). The identification of specific fuzzy elements which are likely to be included need not be exhaustive but need be

sufficient to get a rough grasp. After all these actions, the need of fuzziness can be determined and also a general cognition of specific types of fuzziness underlying in the planned ontology can be obtained.

Inputs:
ExistingCrispOntologies; ExistingFuzzyOntologies; DomainOrScopeOfOntology;
Outputs:
ExistingOntologiesToBeReused; FuzzyElementsToBeReused; CrispElementsToBeReused;
DefinedFuzzyElements;
Functions:
ReusingExistingOntologies (ExistingCrispOntologies, ExistingFuzzyOntologies, DomainOrScopeOfOntology): checking if existing crisp and fuzzy ontologies could be reused, returning ExistingOntologiesToBeReused;
FuzzyElements (ExistingOntologiesToBeReused): checking whether selected ontology elements to be reused are fuzzy or not;
CorrectFuzzyElements (FuzzyElementToBeReused): refining fuzzy elements which are inherited from existing fuzzy ontologies to fit in the domain or scope of the planned ontology;
DefineFuzzyOntologyElements: defining fuzzy ontology elements from scratch to represent corresponding fuzzy related information.
Main:
If ReusingExistingOntologies (ExistingCrispOntologies, ExistingFuzzyOntologies, DomainOrScopeOfOntology) == True then
 if FuzzyElements (ExistingOntologiesToBeReused) == True
 then *CorrectFuzzyElements* (FuzzyElementToBeReused)
 else *DefineFuzzyOntologyElements*
 else *DefineFuzzyOntologyElements*

Figure 18. The flow of step 3, 4, and 5.

Phase 3: Determine fuzzy related information

Since research on methodologies for building crisp ontologies is quite mature, the default setting for the result of step 2 is true which denotes that fuzziness is required in the ontology design. Hence, the main focus of step 3 is put on determining fuzzy related information. Following the step 2, a better understanding of vague information present in the domain could be achieved. In this step, information that really has vague meanings could be identified to a greater extent. A distinction between precise and vague information can be established which could provide valuable inputs for further definitions. Based on the results obtained in this step, the knowledge base in the intended domain could be split into two parts: precise and fuzzy related information. With a clear awareness of the differentiation, ontology engineers could provide different treatments tailored for precise information or fuzzy related information in a well-defined manner.

Phase 4: Consider reusing existing ontologies

Checking existing ontologies relevant to the domain or scope of interest and determining their reusability are the main tasks defined in this phase. Reusing existing resources can give a lot of credits for the ontology design. Mainly, benefits brought by reusing existing ontologies are two-fold: 1) reducing the workload of designing ontologies and saving the design time, and 2) enabling interoperability and compatibility with other applications which commit to the same ontologies. It is worth noting that here existing ontologies refer to not only crisp ontologies but also fuzzy ontologies. Existing fuzzy ontologies are firstly considered and included into the list to check for reusability. It is worth noting that compared with crisp ontologies, existing fuzzy ontologies are fewer and more difficult to navigate. To the best of our knowledge, there is not such a database or hub dedicated to publishing fuzzy ontologies. However, traditional ontology resources, such as W3C wiki [127], Swoogle [128], webpages, domain relevant documents, project documentations, and academic publications, could be visited for existing fuzzy ontologies. For instance, to find existing fuzzy ontologies for recognition of human behavior, a web search using keywords "fuzzy ontology for human behavior recognition" could bring some useful information, such as the source link to an existing fuzzy human behavior ontology (<http://users.abo.fi/ndiaz/public/FuzzyHumanBehaviourOntology/>) and many research papers on fuzzy human behavior ontologies. With the existing fuzzy ontology resources, ontology engineers and domain experts should further examine their relevance to the target domain. Fuzzy ontology elements, which provide approximation and modeling for similar vagueness, could be inherited. In addition, crisp ontology elements defined in existing fuzzy ontologies could also be useful if they are considered as relevant to the target modeling information. This extension of introducing fuzzy ontologies into the existing ontology base can increase the possibility to reuse ontological elements in the ontology design. In this way, reusability of existing ontological resources could be maximized. Apart from existing ontologies, non-ontological resources, such as literal classifications and domain specifications, can also be used to extract useful terminologies and hierarchies. Depending on the fuzziness of existing ontologies which are selected as candidates to be reused, different actions are defined to process crisp or fuzzy elements in order to integrate existing ontology elements into the intended ontology. Figure 18 illustrates the specific treatment to ontology elements that could be reused in terms of fuzziness.

Phase 5: Reuse fuzzy ontology elements

The answer to that whether existing ontologies could be reused could become clear after step 4. If an or several existing ontologies are analyzed to be useful in the ontology design, a fine-grained check should be made on those potential ontologies. The check-up is focused on

inspecting whether selected ontology elements from existing ontologies are fuzzy. Three different kinds of check results may be got: 1) only crisp ontology elements, 2) only fuzzy ontology elements, and 3) both crisp and fuzzy ontology elements could be reused in the planned ontology. If only crisp ontology elements from existing ontologies are identified as useful, then it leads to step 7 which will be specified in subsection Phase 7. Taking into account vague information in the domain of interest, existing fuzzy ontologies might have already provided similar specifications and corresponding modeling to those impreciseness and vagueness. Thus, some fuzzy ontology elements could be picked out from existing ontologies and be potential elements to be reused in the planned ontology. If the check result falls into this case, then further inspection and correction on those fuzzy ontology elements should be made which are explicitly defined as step 6. If the check result is the last case, then both phase 6 and 7 should be activated.

Phase 6: Correct fuzzy ontology elements

In this phase, the involvement of domain experts is required to correct fuzzy ontology elements which are inherited from existing fuzzy ontologies. Specifications and modeling for vagueness provided by existing fuzzy ontology elements may not guarantee a perfect fit to capture the information that is identified as vague in the domain of interest. Therefore, fuzzification for ontology elements should be refined to accommodate the target ontology requirements. For instance, a fuzzy data type *YoungAge* defined in an existing fuzzy ontology O_1 is considered to be reused in the planned ontology O_2 . However, the fuzzy definition for the data type *YoungAge* with range restricted by a leftshoulder membership function [129] $ls(0,90,10,30)$ is identified by domain experts as a mismatch to the vague information '*people aged from 10 to 40 could be regarded as young people*' in O_2 . Based on information provided by domain experts, the fuzzy data type *YoungAge* could be reused in O_2 with a corrected fuzzy set, such as $ls(0,90,10,40)$. It is worth noting that to model the same piece of vague information, different solutions which include different fuzzy ontology elements can be available. To choose the most suitable one from existing modeling is also considered in this phase. Taking the same piece of vague information '*people aged from 10 to 40 could be regarded as young people*' as an example, the vagueness in the definition of young age can be captured using different solutions. One is described previously using a fuzzy data type *YoungAge* to express the vagueness in the definition of young age. Another possibility is to define a fuzzy modifier [129] which could be a function $very=ls(0,90,10,40)$ and use this fuzzy modifier to restrict the property (*isClassifiedAs*) between concept *People* and *YoungPeople*. Therefore, the vague information can be expressed as *People* (and $very(isClassifiedAs)$ *YoungPeople*) or *People* (and *hasAge* *YoungAge*). With activities undertaken

in this phase, existing fuzzy ontology elements can be corrected to ensure an accurate approximation to information which has a vague meaning present in the intended domain or application.

Phase 7: Define fuzzy ontology elements

The output of phase 3, which is a comprehensive understanding of the distinction between fuzzy related information and crisp information, could be regarded as a valuable input in this phase. The goal of this phase is to define different fuzzy ontology elements to provide correct approximations to the nature of vague and imprecise information in the domain. Tight collaborations between domain experts and ontology engineers are needed in this phase. Domain experts are required to provide a clear and specific definition/quantification for vague information based on their expertise or historical statistics. Fuzzification, such as membership functions and certain degree, set by domain experts can reflect imprecise and vague information. Ontology engineers should model vague information by means of fuzzy ontology elements, such as fuzzy concepts, fuzzy relations, and fuzzy data types, in a well-organized manner. The procedure to define fuzzy ontology elements is essentially in line with activities defined in crisp ontology development methodologies, such as enumerating (fuzzy) concepts, building the hierarchy, establishing (fuzzy) relations, and defining specific (fuzzy) data types. However, the significant difference between building fuzzy ontology elements and crisp ontology elements is to accurately capture the vagueness in the specifications and represent it using fuzzy sets. The vagueness and its interpretation of fuzzy degrees need to be precisely modeled based on context, namely, particular knowledge domain or scope. Therefore, domain experts play an important role in this stage. Though there might be just a very small amount of vague information present in the whole domain of interest, to model them associated with fuzzy logic is a key task in the whole development process. Up to this point, all precise and vague information could be correctly addressed and modeled by means of corresponding fuzzy elements within the fuzzy ontology.

Phase 8: Define crisp ontology elements

This phase focuses on dealing with certain knowledge in the domain. Apart from fuzzy related information, the rest of knowledge base in the domain is defined as crisp ontology elements depending on their specific attributes. Activities defined in conventional ontology development methodologies could be applied in this phase to model crisp information. For instance, taking the method proposed in [116] as an example, to enumerate important terms and organize them in a hierarchical manner could be the first step in this phase. To develop the class hierarchy, three approaches can be followed: 1) top-down (starting with the most general

concepts and detailing them to a fine-grained manner), 2) bottom-up (defining the most specific concepts and generalizing them to a higher level), and 3) combination (a mix of the top-down and bottom-up approaches). Relationships could be defined to link different concepts. Other crisp ontology elements, such as data properties, axioms, instances, are also developed in this phase. Up to this point, all elements that form the fuzzy ontology have been defined. The conceptual model for the fuzzy ontology has been completed.

Phase 9: Formalization

A certain language should be selected to formalize the designed ontology into a machine-readable format. Classical ontology languages might not be suitable to express vagueness and imprecision defined in fuzzy ontologies [129]. Hence, different formalism languages have been developed to support the representation of fuzzy ontologies. Syntax and semantics of RDF are extended to support real number on the interval $[0,1]$ to express the certain degree of subject, object, and predicate [130]. A set of fuzzy extensions of DLs [55], could also be adopted to enable the transformation from fuzzy ontology elements to a standard formalization. Besides, Bobillo *et al.* [24] presented a concrete methodology to formalize fuzzy ontologies using OWL 2 annotation properties. Fudholi *et al.* [131] put forward to represent fuzzy ontology elements by means of rules formulated in SWRL. The SWRL-based approach is easy to be used despite it considerably increases the number of rules and limits the scalability of fuzzy ontologies.

It is worth noting that different fuzzy ontology formalism languages vary from each other in terms of characteristics and capabilities they hold. There is not a standard mechanism to evaluate different formalism languages because they have different strengths and weaknesses with regard to representing specific ontology elements. For example, fuzzy data types are not supported by the fuzzy description logic f-SHIN [132] and the SWRL-based approach while they can be easily expressed by fuzzy OWL 2 annotations. Therefore, a specific formalism language should be chosen according to specific fuzzy ontologies' requirements to enable fuzzy expressions.

Phase 10: Validation

The success of creating a fuzzy ontology is subject to the validation result. The designed ontology should go through a thorough check to ensure it has represented the intended model of the world. In this phase, the designed ontology needs to be validated in terms of several features as follows:

- **Correctness.** The developed ontology should be able to accurately reflect information that is included in the target domain. A clear borderline between the crisp information and the fuzzy related information is established in the ontology. Accordingly, crisp and fuzzy information is correctly modeled. Particularly, with a focus on fuzzy elements, it is necessary to ensure that real vague meanings in the domain have been correctly captured, understood, approximated, and treated in the ontology.
- **Consistency.** Local inconsistency in the ontology network should be checked. This feature could be automatically checked by some fuzzy ontology reasoners, such as fuzzyDL reasoner [66], and DeLorean [133]. The consistency issue exists in mainly two aspects: the structure level and the content level. In terms of the structure-based consistency, inclusions of constructors, such as owl:disjointWith and rdfs:subClassOf., should be ensured to avoid any conflicts in the ontology hierarchy. Basic observations should be made on the ontology statements to check if any of them contains controversial definitions for the same specification. In this way, the content-based consistency could be guaranteed.
- **Completeness.** The completeness feature ensures that the designed ontology has been able to cover all the aspects of information that belongs to the target domain. It could provide a complete representation of the real-world knowledge. With a focus on fuzzy related information that is identified by domain experts as significant in the domain, it is a must to ensure that vagueness has been fully captured and included in the fuzzy ontology.
- **Rationality.** The inclusion and quantification for fuzziness, such as fuzzy set and certain degree, make sense to get a good approximation to real information that has vague meanings. A common agreement on the designed treatment for vague information between domain experts and ontology engineers should be achieved.
- **Understandability.** The nomenclature for ontology elements should be easily understandable to all stakeholders, including domain experts, ontology engineers, and ontology users. The naming mechanism used in the ontology should be easy, self-explanatory and intuitive. Understandability could strengthen the ease of use of the designed ontology and promote its usability.
- **Conciseness.** Conciseness is also a significant criterion to be considered to evaluate the quality of ontology. Ontology terms are expected to express the most by using the least number of words. To model the same domain of interest, a lightweight and concise ontology is usually preferable than a heavy one under the condition that they cover the

same knowledge base. Redundancies in the ontology will increase the volume of the ontology and applicable complexities as well.

In general, the aforementioned properties, except consistency, are subjectively examined by humans who have been involved in the development process, including domain experts, ontology users, and ontology engineers. To minimize the side effect of subjectivity in the validation process, it is better to involve as many people as possible, such as another group of domain experts and ontology developers, in verifying the developed ontology. The consistency feature of the developed ontology is usually evaluated by an existing fuzzy ontology reasoner.

Phase 11: Documentation

In this stage, documentation to introduce the engineering principles of the designed ontology, including descriptions for different ontology elements, design details, the method of usage, and maintenance *etc.*, should be written up. As communicable materials to the public, the documentation should be concise, illustrative, understandable, and comprehensive so that non-experts (*e.g.*, ontology users) can easily identify the potential usage of this ontology in their own applications by looking up the document. Besides, enabling the developed ontology as open source to the ontology community is another step forward. Open access to the ontology can expand its dissemination and increase the possibility of reusability in other projects or applications. In addition, valuable feedback from the ontology community can also be collected and used to make a better revision or maintenance on the ontology development.

4.1.3. Uncertainty Modeling Process for Semantic Technology (UMP-ST)

Carvalho *et al.* [126] proposed a methodology, named the Uncertainty Modeling Process for Semantic Technology (UMP-ST), for modeling probabilistic ontologies based on the MEBN theory. The UMP-ST draws upon related processes for software engineering (*e.g.*, the unified process (UP) [134]), ontology engineering (*e.g.*, the METHONTOLOGY and On-To-Knowledge methodology), and Bayesian network engineering to provide a process tailored to probabilistic ontology engineering.

The UMP-ST is an iterative and incremental process, based on the UP, for designing a probabilistic ontology. The specific phases which it consists of can be seen in Figure 19. The UMP-ST includes all phases of the UP but focuses only on the Requirements, Analysis & Design, Implementation, and Test disciplines. As can be seen from the Figure 19, each phase of the UMP-

ST includes all four disciplines, but the emphasis shifts from requirements in the earlier phases toward implementation and test in the later phases.

Figure 19. UMP-ST, extracted from [126].

Figure 20 shows the specific Probabilistic Ontology Modeling Cycle (POMC). The major outputs from each discipline in a sequential order are depicted in this figure. The POMC cycles through the steps iteratively, using what is learned in one iteration to improve the result of the next. The Requirements discipline (blue box) defines the goals for the probabilistic ontology. In other words, what is willingly achieved by reasoning with the semantics provided by the intended probabilistic ontology? Usually, when designing a probabilistic ontology, one wants to be able to automate a reasoning process which context uncertainty is involved in. The Analysis & Design (green boxes) describes classes of entities, their attributes, how they relate to each other, and what rules apply to them in the intended domain. These definitions are independent of the language that is used to implement the model. The Implementation discipline (red boxes) focuses on mapping the design to a specific language that is both semantically rich and able to represent context uncertainty. This means encoding the classes, attributes, relationships, and rules in the chosen language. In this thesis, the mapping is to PR-OWL. Finally, the Test discipline (purple box) is in charge of evaluating whether the model developed during the Implementation discipline lives up to the expectation from the rules defined during Analysis & Design and whether the results reach the goals elicited during the Requirements discipline. Like several of the ontology engineering processes considered by Corcho *et al.* [135], the UMP-ST fails to cover ontology management, under the assumption that these activities can be imported from another framework.

Figure 20. POMC, extracted from [126].

4.2. The SWARMs ontology

By following the FODM methodology, a common information model, named the SWARMs ontology, is built and presented in this chapter. The design of the SWARMs ontology is a collaborative work with marine experts in the SWAMRs project. It is necessarily noted that fuzzy and probabilistic modeling is application/scenario specific. Therefore, two specific SWARMs scenarios will be presented to demonstrate how the SWARMs ontology can be extended with fuzzy and probabilistic extensions. The FODM and the UMP-ST methodology will be adopted to guide the fuzzy and probabilistic modeling, respectively. For the sake of conciseness, the step-by-step design details are omitted and the final ontology proposal, namely the SWARMs ontology, will be directly described in the following.

4.2.1. The domain and scope of ontology

It is very important to clarify the domain and scope of the target modeling information. With a clear understanding of the purpose and scope, it is possible to define what concepts should be

included or excluded from the SWARMS ontology. In addition, the viability, domain, and objectives of the SWARMS ontology can set requirements for the ontology design and provide an initial idea of the underlying semantics. The SWARMS ontology intends to model all information that is necessarily exchanged between any maritime or underwater vehicles and architecture components (e.g., middleware modules, MMT modules). In the scope of the SWARMS project and its application scenarios (e.g., oil spill detection, plume tracking, and berm construction), the wide range of information could be summarized into four different domains, which include robotic vehicles, mission & planning, environment recognition & sensing, and communication & networking. The purpose of developing the SWARMS ontology is to provide a formal representation of all four domain-specific information so that context heterogeneity can be abstracted and a common understanding can be achieved by vehicles and any SWARMS architecture components or entities. In addition, it is necessary to ensure extensibility of the SWARMS ontology so that it could be tailored to different scenarios with application extensions (e.g., fuzzy extensions or probabilistic extensions).

4.2.2. Design requirements

In order to achieve the purpose and ensure an appropriate outcome, the SWARMS ontology should meet a set of requirements. These requirements can be grouped by non-functional and functional requirements.

Non-functional requirements

Non-functional requirements are general requirements or aspects that an ontology should fulfill for the sake of modeling quality. They also refer to those ontological principles that guide the design process. According to the NeOn methodology, interoperability, modularity, reusability, and extensibility, are the important characteristics that an ontology should offer. Specifically, the SWARMS ontology should ensure interoperability between heterogeneous vehicles or software components in terms of syntax and semantics. With the interoperability feature, the SWARMS ontology will enable vehicles with a common understanding of the information exchanged between themselves. The SWARMS ontology should also be modular so that minimum coupling and maximum cohesion could be achieved. In addition, reusability is a must for the SWARMS ontology, thus supporting that its portions could be re-engineered in different domains or applications. The SWARMS ontology should be able to be extended with application enrichment. With the extensibility, it could be stretched in width in order to suit different applications or scenarios. In addition, the SWARMS ontology should be formalized following two more non-

functional requirements [65]: (1) Understandability. The nomenclature of the SWARMS ontology should be easily understandable to all stakeholders, e.g., ontology engineers, marine experts, end users, and operators. The naming for the SWARMS ontology elements should be self-explanatory and reveal an intuitive meaning. (2) Conciseness. To model the same domain of interest, a lightweight ontology is usually preferable to heavier ones. The SWARMS ontology should use the least number of words to express the most without redundancies so as to decrease the complexity of the design process.

Functional requirements

The SWARMS ontology can fulfill a set of functional requirements, namely, content-specific requirements, in order to fully represent information within the target scope. The functional requirements are grouped as follows.

- The SWARMS ontology must provide the mission & planning modeling. Two levels of abstraction of the mission & planning can be described by the SWARMS ontology. Firstly, the high-level planning that allows the user to describe different tasks regarding operations performed by a set of robotic vehicles without specifying the exact actions that each robotic vehicle needs to perform. The output of the high-level planning is a global mission plan consisting of the tasks that the swarm of robotic vehicles needs to perform. Secondly, low-level planning that is carried out at the robotic vehicle level and includes generation of waypoints, actions, and other similar low-level tasks. The output of the low-level planning is a vehicle plan. The mission & planning procedure needs to be decomposed and well represented in the SWARMS ontology so that vehicles can share tasks/operations/actions and understand them in the same manner so that cooperation and coordination could be fostered.
- The SWARMS ontology must provide a well-defined classification for the robots and vehicles that are used in the different missions and their attributes. Any information that could be useful for operators to understand the vehicles and their conditions is modeled in the SWARMS ontology. Different properties used to describe vehicles, such as motorized, propelled, non-motorized, speed, position, battery level, equipment, capabilities, and sensors onboard, are modeled with semantic annotations in the SWARMS ontology.
- The SWARMS ontology must provide an abstraction for communication & networking in the SWARMS architecture. It must describe the communication links available in SWARMS

architecture to transfer information from the command and control station (CCS) to the vehicles and backward. In addition, it must provide modeling of the protocol and types of messages that can be transmitted within the SWARMS system.

- The SWARMS ontology must support the environment recognition & sensing modeling. Robotic vehicles involved in a mission should have a complete picture of the underwater environment so that they could better adapt to it accordingly. Thus, the SWARMS ontology provides a good representation of the underwater environment. Any information, that is defined targeting to characterize the environment, its recognition, and associated sensing, is properly modeled in the SWARMS ontology. For instance, sensors play a very significant role in sensing the environment and producing useful context data to represent it. The environment is defined by a set of main concepts, which are specified by particular properties that define the surroundings of the location where a mission or task takes place involving robotic vehicles. A variety of environmental properties, such as water salinity, conductivity, temperature, and currents, are formalized in the SWARMS ontology.
- The SWARMS ontology must model context uncertainties and support for uncertainty reasoning. The harsh maritime and underwater environment typically introduces uncertainties in context data, particularly in such data obtained by sensors or other sensing instruments. The SWARMS ontology can provide a suitable representation for context uncertainties for the sake of completeness and comprehensiveness. In addition, the uncertainty annotations provided by the SWARMS ontology are useful for further reasoning. In other words, the SWARMS ontology can enable the ease of applying uncertainty reasoning in order to generate more useful information. It is noted that uncertainty is classified into different types as described in section 2.1.2. In the SWARMS project, uncertainty features, inaccuracy, incompleteness, and randomness, are the major issue aiming to be addressed. In addition, the rest of uncertainty features, vagueness and ambiguity, will be secondarily considered and investigated for a potential solution.

4.2.3. Description of the SWARMS ontology

In order to fulfill all the requirements presented in section 4.2.2, the SWARMS ontology is designed as a network of ontologies. The overview of the structure of the SWARMS ontology can be seen in Figure 21. As depicted in Figure 21, the SWARMS ontology mainly consists of four domain-specific ontologies which model common concepts and aspects of certain domains, including mission & planning, environment recognition & sensing, robotic vehicle, and

communication & networking. All these domain-specific ontologies are interlinked through a core ontology. In addition, the SWARMS ontology could be enriched with application extensions in order to suit certain use cases and scenarios which context uncertainty is involved in. In the following sections, the specific design for different domain-specific ontology model and their relationships in the SWARMS ontology will be presented. The detailed description for the SWARMS ontology can be seen in Appendix A.

Figure 21. The general architecture of the SWARMS ontology.

Core ontology

The SWARMS ontology aims to model information mainly related to four domains. In order to link domain-specific ontologies and provide a coherent representation, a core ontology, shown in Figure 22, is presented. In Figure 22, the main concepts from the mission & planning ontology are depicted in white while concepts from the environment recognition & sensing domain are marked in gray, yellow, and red. In addition to that, ontology elements from the communication &

networking domain and concepts from the robotic vehicles domain are displayed in orange and green, respectively.

As illustrated in Figure 22, services or capabilities (abstracted as the Concept Service in the mission & planning ontology) which are necessary to fulfill any task (modeled as the concept Task in the mission & planning ontology) are linked with Asset (defined in the robotic vehicles ontology) by using a pair of inversive relationships, namely, *providedBy* and *contributes*. The mission & planning ontology could also be interrelated with the robotic vehicles ontology at a lower abstraction level. Specifically, the concept VehicleLevelTask could be linked with RoboticVehicle by object properties (*assignedTo* and *allocatedTo*). In addition, a pair of inversive object properties, namely, *performedBy* and *canPerform*, are defined to describe the relationships between the concept Action and the concept RoboticVehicle. The concept Sensor from the environment recognition & sensing ontology and the concept CommunicationLink from the communication & networking ontology are modeled as subclasses of the concept System in the robotic vehicles ontology. The main concept Vehicle from the robotic vehicles ontology is subsumed into the concept ManmadeObject from the environment recognition & sensing ontology. With all the aforementioned ontology statements, the core ontology is able to glue different ontology elements from independent domain-specific domains and enable the building of a network of ontologies.

Figure 22. A representation of the overall structure of the core ontology.

Robotic vehicles ontology

The Robotic Vehicles ontology models the robots and vehicles that are used in the different SWARMS missions (in the water domain only). Some important elements that capture the model are:

- Mobile robots are vehicles and robots (polymorphism);
- Robots are either autonomous robots, automated robots or remotely piloted robots (disjunction);
- Vehicles are either underwater vehicles or surface vehicles (disjunction);
- Vehicles are either motorized or propelled by the environment/unmotorized (disjunction).

Figure 23 shows the taxonomy of vehicles and robots used in SWARMS. A vehicle models any platform in or by which someone travels or something is carried or conveyed. Vehicles can travel underwater or operate on the surface of the water (e.g., Vessel). Motorized vehicles (e.g., ROV, USV, Vessel) have propulsion systems onboard, unlike unmotorized vehicles (e.g., UnderwaterGlider, SurfaceGlider, Buoy). A robot is a mechanical device that is capable of performing a variety of complex tasks on command or by being programmed in advance. It

operates by remote control, automatically or autonomously. In a general view, this also includes humanoid and service robotics. A robotic vehicle is a physical object designed by a human agent to provide a service by acting here on the water domain. Remotely piloted robots (e.g., ROV) are robots that are piloted by one or several human operators that use sensory feedback. Automated robots are robots not remotely piloted (in their main role) that are able to act as an automation but not to adapt to changes in the environment and/or to follow scripted plans. Autonomous robots (e.g., AUV, ASV) are robots neither remotely piloted nor automated (in their main role) that are able to perform high-level tasks and adapt to changes in the environment and operations with limited human intervention. Underwater robots (e.g., AUV, ROV) are underwater and robotic vehicles. Surface robots (e.g., ASV, USV) are surface and robotic vehicles. In this robotic vehicle domain-specific ontology, the UnderwaterGlider and SurfaceGlider can glide using density-volume changes without a propulsion system and they are subsumed into the AUV and ASV class, respectively.

Figure 23. Robotic Vehicle taxonomy.

Mission & planning ontology

The mission & planning ontology provides a general representation of the whole mission composition and planning procedure and of the low-level planning at a vehicle level as illustrated in Figure 24. A mission is defined as a set of goals to be performed by a swarm of vehicles (e.g., AUV, ROV, and USV) where each goal represents an objective to be achieved. Goals can be

divided into subgoals. A goal is achieved by executing 1 to n tasks. These tasks can be of 3 types: operator level, vehicle level or high-level tasks. An operator level task is manually carried out by an operator. A vehicle level task can be carried out by one single vehicle (AUV, ROV or USV) whereas a high-level task is an assembly of tasks (operator level, vehicle level and/or high-level) that will be carried out by a swarm of vehicles. Tasks require capabilities to be performed (e.g. bathymetric sensors, H₂S sensor or camera), a minimum battery level and have a start and end location. Vehicle level tasks lead to a set of actions (e.g., dive, go to waypoint, follow row or communicate status) to be performed by the vehicle.

Hence, a mission plan is abstracted as the sequence of scheduled low-level tasks (operator and vehicle level tasks) that need to be carried out by the swarm of vehicles to achieve a mission, with dependencies between tasks and approximate time duration. In addition, a vehicle plan is modeled as the sequence of actions that need to be carried out by a vehicle to achieve the set of tasks assigned to it in the global mission plan.

Figure 24. The mission & planning ontology.

Environment recognition & sensing ontology

This domain-specific ontology targets to characterize the environment, through recognition and sensing, where maritime or underwater missions will be carried out. The environment can be defined by a set of abstract concepts, which are specified by particularly associated properties,

that define the surroundings of the location where a particular mission or tasks take place involving robotic vehicles. This ontology is structured around a set of three disjoint spatial domain concepts, represented in blue, *i.e.*, Surface, Water Column, and Seabed, each being characterized by multiple properties, some of which similar, *e.g.*, Temperature and where Entities and Landmarks or Features can exist, and be found/recognized, or not, through sensing, in such SWARMS typical environment. Such concepts' data properties, *e.g.*, Temperature and most other sensed characteristics, must always be associated with a position and timestamp, which are typically provided by the vehicle that supplies the respective sensor (s) reading (s) in the characterization process.

Entities can be Biotic, *i.e.* animals or plants, or Manmade objects. Landmarks (or Features) represent all other kinds of objects that can be considered/recognized as landmarks, *e.g.*, big rocks, as well as geological formations on the seabed. Infrastructures are considered usually as landmarks due to their size and construction records, *i.e.*, are registered and mapped, and are also obviously manmade objects (polymorphism). Other objects, besides Infrastructures and objects with Sensors (*e.g.*, AUV, ROV) are also considered, namely since it could be important in a mission or task to find/recognize them, *e.g.* oil pipes or components of an infrastructure.

Moreover, in the representation of the Environment recognition & sensing ontology in Figure 25, the concepts in red indicate the ones that could be important to be recognized after sensing, according to the specific SWARMS mission. In a comparable way, the Sensor concept is represented in yellow since it is the one that clearly is associated with the sensing part of the ontology, being the source of all sensing data, which is used to characterize the respective environment and allows the recognition of entities or objects and landmark or features. Typically, the Sensors are installed on robotic vehicles, but can also eventually be attached to other Manmade objects, including Infrastructures. Manmade objects and Sensors are therefore evidently the interfacing concepts between this domain-specific ontology and the Robotic vehicles ontology.

Figure 25. The environment recognition & sensing ontology.

Communication & networking ontology

This information model, shown in Figure 26, describes the communication links available in SWARMS to interconnect the different agents involved in a global mission execution and supervision, *i.e.*, the ashore control station, support ships, USVs, AUVs, and ROVs.

The communication network should allow the data exchange in different environments (underwater and surface). The underwater environment is especially challenging as the propagation delay is significant, communication links are highly unstable and channel bandwidth is very limited.

In particular, the link between surfaced AUVs, USVs or buoys and the support ship is made through a radio frequency (RF) connection. In addition, satellite communication could be used with the CCS hosting the MMT, when offshore. For the underwater communication, acoustic modems are used with AUVs whereas ROVs are connected to the support ship by cables, which are used to transmit command and control signals between the operator and the ROVs.

Figure 26. Communication & networking ontology.

Application ontology

When dealing with different scenarios (e.g., oil spill detection, plume tracking, berm construction, and corrosion repair), the SWARMS ontology need be enriched with application extensions in order to accommodate application-specific requirements.

To deal with ambiguity and vagueness in specific scenarios, fuzzy annotations can be added in the SWARMS ontology by using fuzzy ontology constructs.

To deal with context inaccuracy, incompleteness, and randomness, PR-OWL constructs can be used to model the uncertainty extensions. In the following, two scenarios are described in order to exemplify how the SWARMS ontology can be extended based on fuzzy ontology and MEBN theory.

4.2.4. Formalization of the SWARMS ontology

Protégé 5.1.0 [136] is adopted to formalize the SWARMS ontology described in the previous section in a semantic format (e.g., OWL and RDF). Protégé is a free, open source ontology editor and a knowledge management system. It provides a graphical user interface to define the structure of an ontology, manage instances, specify rules, *etc.* The interface provided by Protégé for the creation of ontologies can be seen in Figure 27.

Figure 27. Graphical user interface provided by Protégé.

The formalization of the SWARMs ontology in OWL allows a great level of expressivity while producing a model that can be easily shared through the web and thus be open to third party extensions. The proposed SWARMs ontology has been continuously inspected and evaluated by marine experts and ontology engineers along the development process. In addition, the reasoning result using the Pellet [137] reasoner has shown that the proposed SWARMs ontology is consistent. The hierarchy of the proposed SWARMs ontology is shown in Figure 28.

Figure 28. The snapshot of the hierarchy of the SWARMS ontology.

It is worth mentioning that the Semantic Web Rule Language (SWRL) is adopted in the SWARMS ontology to compensate the inability of OWL to express complex rule formations and relations. By adding SWRL rules, the SWARMS ontology can be enhanced in terms of logicity, human-readability, expressivity, and completeness.

4.2.5. Fuzzy extensions: an example of seabed characterization

In the environment recognition & sensing domain-specific ontology, seabed and its different types are characterized. Assuming in a specific scenario, vehicles are assigned to survey

the underwater ground. The size of the inspected seabed region is also of interest for operators in order to get a better understanding of the underwater ground. From the marine experts and operators' point of view, a literal expression for the seabed region size is more preferable, such as large, medium, or small. Thus, there is a need to add some linguistic specifications and characterizations for the seabed region size. However, how to map a seabed region with the explicit numeric area known to a linguistic specification implies vagueness. The borderline between each literal type is blurry and overlap between each type could exist. In this sense, fuzzy extensions can be added to the SWARMS ontology in order to model vagueness inherent to the region size. Following the FODM methodology, a set of fuzzy ontology elements can be defined.

To provide linguistic classifications for the size of seabed regions, three fuzzy data types and four fuzzy concepts are defined with the collaboration of marine experts. Definitions of fuzzy data types which follow the fuzzyDL reasoner syntax [66] and vague information they intend to model are shown in Table 2. Specifications for fuzzy concepts defined in the fuzzy seabed characterization ontology are also presented in Table 3. The expressions for fuzzy concepts follow the syntax of fuzzy Description Logics. In principle, the definition of fuzzy data type aims to provide the corresponding specification for the data format of a fuzzy concept, such as *SmallSize* restricts the numeric size of *SmallRegion* seabed and also generates a specific probability for a seabed area to be classified as small. Thus, a crisp data property, *hasNumericValueSize*, should be defined in order to specify the relationship between fuzzy concepts (Small, Medium, and Large) and fuzzy data types (*SmallSize*, *MediumSize*, and *LargeSize*).

Table 2. Fuzzy data types

Fuzzy data type	Definition	Vague information modeled
<i>SmallSize</i> =leftshoulder (0,1000,50,100)	Denoting that the numeric size of a small seabed region should comply with a leftshoulder membership function leftshoulder (0,1000,50,100).	Seabed with its size ranging from 0-100 m^2 could be regarded as small to some degree. The degree distribution complies with a leftshoulder membership function.
<i>MediumSize</i> =trapezoidal (50,100,150,200)	Denoting that the numeric size of a medium seabed region should comply with a trapezoidal membership function trapezoidal (50,100,150,200).	Seabed with its size ranging from 50-200 m^2 could be regarded as medium to some extent. The degree distribution complies with a trapezoidal membership function.
<i>LargeSize</i> =rightshoulder(0,1000,150,200)	Denoting that the numeric size of a large seabed region should comply with a rightshoulder membership function	Seabed with its size ranging from 150-1000 m^2 could be regarded as large with a possibility. The degree distribution complies with a rightshoulder membership function.

	rightshoulder(0,1000,150,200).	
--	--------------------------------	--

Table 3. Fuzzy concepts

Fuzzy concept	Definition	Vague information modeled
SeabedRegion	Representing the superclass of a set of subconcepts, including SmallRegion, MediumRegion, and LargeRegion.	Seabed region could be described by linguistic variables, such as small region, medium region, and large region.
SmallRegion	Containing a collection of seabed whose size is assigned with the SmallSize fuzzy data type. $SmallRegion \equiv \exists hasNumericValueSize. SmallSize$	Seabed area size ranging from 0-100 m^2 is classified as small by complying with a leftshoulder membership function.
MediumRegion	Containing a collection of seabed whose size is assigned with the MediumSize fuzzy data type. $MediumRegion \equiv \exists hasNumericValueSize. MediumSize$	Seabed area size ranging from 50-200 m^2 is classified as medium by complying with a trapezoidal membership function.
LargeRegion	Containing a collection of seabed whose size is assigned with the LargeSize fuzzy data type. $LargeRegion \equiv \exists hasNumericValueSize. LargeSize$	Seabed area size ranging from 150-1000 m^2 is classified as large by complying with a rightshoulder membership function.

As shown in Table 2 and Table 3, marine experts provide fuzzification for the blurry borderlines between small, medium, and large size using three fuzzy sets, namely membership functions. More specifically, fuzzy sets, which are encased in the fuzzy seabed characterization ontology to describe fuzzy data types can be seen in Figure 29.

In this example, OWL 2 is selected as the formalism language to represent the designed fuzzy extensions. The fuzzy extensions can be seen in Figure 30.

Figure 29. Fuzzy data types for seabed region size.

Figure 30. The overall visualized structure of the seabed characterization fuzzy ontology.

4.2.6. Probabilistic extensions: an example of oil spill monitoring

In this section, an oil spill monitoring scenario is described. This ultimate goal of this case study is to illustrate how the probabilistic ontology can be able to represent specific scenarios and associated uncertainties in a principled formalism and can support the MEBN reasoning.

Description of the oil spill monitoring scenario:

The proactive detection of oil spills is an important means to minimize the damage of spills to the marine environment. Thus, oil spill detection is considered as one of use cases in the SWARMs project. Assuming that a set of SWARMs vehicles could obtain different types of contextual data, with this data, they can collaboratively detect the occurrence of oil spills. After detecting oil spills in a specific marine region, it is significant to predict the severity level of the spilled region so that remedial measures (*e.g.*, clean-up, containment) could be taken accordingly. The detected oil spill can be characterized by different severity level, *e.g.*, high, medium, and low. Initially, a few simple criteria are selected to estimate this severity level, including the thickness of spills, the estimated size of spills, the weather condition, and the currents. Estimation of this severity level is subject to the aforementioned context. In addition, the severity level of the specific area could be influenced by different spills concurrently occurred in the same area.

Following the UMP-ST methodology, a probabilistic ontology can be built to formalize information in this scenario. The process of creating this probabilistic ontology and the detailed description of this ontology are provided in the following.

Requirements

The objective of the requirements discipline is to define the objectives that must be achieved by creating a formal representation of this scenario-specific semantics and reasoning with it.

For this discipline, it is significant to come up with questions that the probabilistic ontology is expected to answer (*e.g.*, the queries to be posed to the context reasoner component being designed). Similar to the development of traditional ontologies, there are basically two types of requirements: functional and non-functional. In this case study, functional requirements are the only emphasis because they are statements related to what the probabilistic ontology should provide, what features it should have, what knowledge it should cover. Keeping this in mind, the following set of goals/queries/evidence is elicited.

1. Identify in which condition the detected spill is severe and it needs immediate measures.
 - a) Is the spill a lot in terms of volume and size?
 - i. Measure if the spill is thick.
 - ii. Survey the spilled region and estimate if the polluted area is large.
 - b) Is the spill likely spreading fast?
2. Identify criteria of affecting the potential spread speed of the spill.
 - a) What is the weather like when and after the spill occurs?
 - i. Measure the weather features, *i.e.*, wind speed, wind direction, temperature and characterize the weather into two general cases: clement and inclement.
 - ii. Look for historical data on the trend between weather conditions and occurrence of spills.
 - b) How does the water current affect the spread speed of the spill?
 - i. Measure the water current (fluctuates frequently, strongly, *etc.*) and characterize if it is strong or weak.
 - ii. Study statistics of environment conditions when spills occurred.

Analysis & Design

With the goals and potential solutions to achieve them clarified, it is time to model the entities, their attributes, relationships, and rules to make that happen. In essence, this is purpose defined in the analysis & design discipline. In other words, the definition of the semantics of the probabilistic model is the major objective of this discipline.

Figure 31 illustrates the semantics of the oil spill monitoring model. A Spill has a specific location (Region) and the specific location is linked to it by a relation *locatedIn*. In addition, a Spill has features: it has an estimated size (EstimatedSize) and it has a specific Thickness. In addition, a Spill has a potential SpreadSpeed which is determined by Currents and Weather. The SeverityLevel is an important feature of a Spill which is the information of interest to marine experts. All the aforementioned features of a Spill affect the SeverityLevel. The SeverityLevel is linked to them via a relationship *dependsOn* whose attribute is set as transitive. More detailed description of those reference factors is provided in the following.

- SpreadSpeed. This concept includes two types of velocity, namely, Fast and Slow, to describe the possible spreading speed of the detected oil spill. It is asserted that the spreading speed depends on two features: Weather and Currents.
- Weather. It conceptualizes two kinds of weather conditions: Clement and Inclement. Spills often occur in an inclement weather and spread with more possibilities in such weather.
- Currents. It describes that currents in the spilled region fluctuate strongly or weakly.

- EstimatedSize. It is a characteristic of spills and it represents the estimated coverage range of spills. Specifically, it is divided into two kinds including Large and Small.
- Thickness. It defines a measuring unit to estimate the quantity of spilled oil, such as Thick and Thin.

Figure 31. Entities, their attributes, and relations for the oil spill monitoring model.

The probabilistic rules for this model include:

1. If the weather when the spill occurs is inclement, then it is more likely that the spill will spread fast.
2. If the sea water fluctuates quite frequently and strongly, then it is more likely that the spill will have a fast spreading speed.
3. If 1 or 2, then the spill is more likely to spread fast.
4. If the spill spreads fast, then it is more likely the emergency level of the spill is high.
5. If the quantity of the spilled oil is a lot, then it is more likely that the severity level of the spilled region is high.
6. If the spilled oil covers a large area, then it is more likely the emergency level of the spill is high.

7. If 4, 5, or 6, then the spill is more likely to threaten the sea environment and the immediate measures are necessary.

Implementation

Once the analysis & design is finished, it is time to implement the designed model in a specific language. This section describes how to model the oil spill monitoring scenario in PR-OWL using UnBBayes. It is a fact that the main purpose of this phase is to map the entities, their attributes, and relations defined in the previous stage to PR-OWL which uses essentially MEBN terms. This mapping requires ontology engineers to have deep knowledge on PR-OWL syntax and the relations between OWL and PR-OWL formalisms. This implies that the mapping step is not an easy task and also it is error-prone due to its complexity. Fortunately, Carvalho *et al.* [70] [71] developed UnBBayes, a probabilistic network framework written in Java. It has both a GUI and an API to model probabilistic ontologies in PR-OWL based on MEBN theories and reason about the MEBN theory. In this case study, the UnBBayes GUI is adopted to make this mapping. It allows to create the probabilistic ontology in an MEBN theory and store it in the PR-OWL language. For the sake of conciseness, the detailed steps to create this probabilistic ontology using UnBBayes are not presented in this thesis. For more information about how to use the UnBBayes GUI to model probabilistic ontologies, it can be referred to [70] [138].

Figure 32. The oil spill monitoring probabilistic ontology formatted in an MEBN theory.

The MEBN model, that represents the final probabilistic ontology for the oil spill monitoring, is created based on the requirements and rules identified in previous stages. For instance, the rule 3 indicates that there is a dependence between Weather(rgn), Currents(rgn), and SpreadSpeed(oil).

As can be seen in Figure 32, the MTheory consists of seven MFrag, including Thickness, EstimatedSize, Weather, Currents, Location, SpreadSpeed, and SeverityLevel. Each MFrag represents knowledge for a specific entity and its local probability distribution (LPD) [139]. In each MFrag, the resident RVs are shown as yellow rounded rectangles, the input RVs are shown as gray trapezoids, the context RVs are shown as green pentagons. Specifically, the Thickness, EstimatedSize, Weather, Currents, and Location MFrag provide modeling of variable Thickness, EstimatedSize, Weather, Currents, and Location and their prior probability distribution, respectively. The two main goals described in previous requirements are defined in the SpreadSpeed and SeverityLevel MFrag. The SpreadSpeed and SeverityLevel MFrag represent the local probability distribution of variable SpreadSpeed and SeverityLevel under the influence of their causal variables, such as Weather, Currents, and Thickness. The definition of the local probability distribution for all resident nodes is a very important step in constructing this probabilistic ontology.

The details and explanations of all MFrag and all resident nodes and their respective LPDs of the probabilistic ontology are presented in the following. Notably, the MFrag are presented from the less dependent to the more dependent. More specifically, an MFrag will only be described after all its dependent MFrag have been described. It is consistent with the order of the dependency explanations given previously. Also, all the LPDs defined in this probabilistic ontology are notional only. In other words, no real data or statistics were used and all the probability information in this model is directly provided by marine experts based on their experiences. Therefore, before this model can be used in production, a few more iterations are necessary in order to make sure these notional probabilities are correct.

Figure 33 presents the Thickness MFrag. The RV Thickness is associated with class Spill. Figure 34 presents the LPD for the RV. The assumption behind this LPD is that the spill which is detected is more likely to be thick.

Figure 33. MFrag Thickness.

Figure 34. LPD for Thickness(oil).

Figure 35 presents the EstimatedSize MFrag. The RV EstimatedSize is associated with class Spill. Figure 36 presents the LPD for the RV. The assumption behind this LPD is that the spill which is detected has the same probability to be large or small.

Figure 35. MFrag EstimatedSize.

Figure 36. LPD for EstimatedSize.

Figure 37 presents the Weather MFrag. The RV Weather is associated with class Region. Figure 38 presents the LPD for the RV. The LPD indicates that the oceanic weather has a prior probability of 70% to be clement and a prior probability of 30% to be inclement.

Figure 37. MFrag Weather.

Figure 38. LPD for Weather.

Figure 39 presents the Currents MFrag. The RV Currents is associated with class Region. Figure 40 presents the LPD for the RV. The LPD indicates that the current has a prior probability of 70% to be strong and a prior probability of 30% to be weak.

Figure 39. MFrag Currents.

Figure 40. LPD for Currents.

Figure 41 presents the Location MFrag. This MFrag is a reference MFrag which specifies that the relationship between class Spill and class Region is a Spill is located in a specific Region.

Figure 41. MFrag Location.

Figure 42 presents the SpreadSpeed MFrag. The local probability distribution of SpreadSpeed, shown in Figure 43, follows UnBBayes MEBN expressive grammar for defining location probability distributions. More information can be referred to [70]. Annotation of this probability information using PR-OWL constructs can be seen in [71].

The MFrag must meet the context constraints (the inspected pollution must be a type of Spill, the inspected area belongs to the Class Region, and this inspected spill locates in the inspected region) in order to be instantiated. The assumption behind the LPD is that the spread speed of the detected spill is more likely to be fast if the weather is inclement and the currents are strong.

Figure 42. MFragment SpreadSpeed.

```

if any rgn have ( Weather = Clement & Currents = Strong ) [
    Fast = 0.5,
    Slow = 0.5
]
else if any rgn have ( Weather = Clement & Currents = Weak ) [
    Fast = 0.1,
    Slow = 0.9
]
else if any rgn have ( Weather = Inclement & Currents = Strong ) [
    Fast = 0.9,
    Slow = 0.1
]
else [
    Fast = 0.5,
    Slow = 0.5
]

```

Figure 43. LPD for SpreadSpeed.

```

<hasProbDist>
<DeclarativeDist rdf:ID="SpreadSpeed_Table">
<hasDeclarationrdf:datatype="http://www.w3.org/2001/XMLSchema#string">
if any rgn have ( Weather = Clement & Currents = Strong ) [ Fast = 0.5, Slow = 0.5 ] else if any rgn
have ( Weather = Clement & Currents = Weak ) [ Fast = 0.1, Slow = 0.9 ] else if any rgn have (
Weather = Inclement & Currents = Strong ) [ Fast = 0.9, Slow = 0.1 ] else [ Fast = 0.5, Slow = 0.5 ]
</hasDeclaration>
<isProbDistOf rdf:resource="#Domain_Res.SpreadSpeed"/>
</DeclarativeDist>
</hasProbDist>

```

Figure 44. The LPD of SpeedSpeed represented in the PR-OWL syntax.

Figure 45 presents the SeverityLevel MFrag. Figure 46 presents the LPD for the RV SeverityLevel(rgn). The assumption behind the LDP is that if the detected spill is thick, the spread speed is fast, and the estimated size of the spill is large, then it is more likely the situation is emergent and it needs quick remedial measures.

Figure 45. MFrag SeverityLevel.


```

if any oil have ( Thickness = Thick & EstimatedSize = Large & SpreadSpeed = Fast) [
    High = 0.9,
    Low = 0.1]
else if any oil have ( Thickness = Thick & EstimatedSize = Large & SpreadSpeed = Slow ) [
    High = 0.7,
    Low = 0.3]
else if any oil have ( Thickness = Thick & EstimatedSize = Small & SpreadSpeed = Fast) [
    High = 0.7,
    Low = 0.3]
else if any oil have ( Thickness = Thick & EstimatedSize = Small & SpreadSpeed = Slow) [
    High = 0.5,
    Low = 0.5]
else if any oil have ( Thickness = Thin & EstimatedSize = Large & SpreadSpeed = Fast) [
    High = 0.6,
    Low = 0.4]
else if any oil have ( Thickness = Thin & EstimatedSize = Large & SpreadSpeed = Slow) [
    High = 0.3,
    Low = 0.7]
else if any oil have ( Thickness = Thin & EstimatedSize = Small & SpreadSpeed = Fast) [
    High = 0.2,
    Low = 0.8]
else [
    High = 0.1,
    Low = 0.9]

```

Figure 46. LPD for SeverityLevel.

With the aforementioned seven MFragS, the MTheory could collectively model the unique joint probability distribution for the SeverityLevel entity. The MTheory represents an infinite number of possibilities, thus, it can be instantiated to specific scenarios and make inferences based on observed evidence correspondingly.

Test

Test plays an essential role in the UMP-ST methodology. According to Laskey and Mahoney [140], the test discipline goal is to find flaws and areas for improvement in the model. After the elicitation review (review the model documentation, analyze if all the requirements were addressed in the final model, make sure all the rules defined during the analysis & design stage were implemented, validate the semantics of the concepts described by the model, etc.), the probabilistic ontology for oil spill monitoring is verified. With regard to evaluating the probabilistic ontology with different scenarios, it actually shifts to the MEBN reasoning which is the focus of section 5.2.3 and chapter 6. Therefore, testing the probabilistic ontology with different scenarios is not provided in this section and it will be detailed in section 5.2.3 and chapter 6.

4.3. Uncertainty quantification in mathematics-embedded ontologies using Stochastic Reduced Order Model

This chapter will present the first application of Stochastic Reduced Order Model (SROM) in quantifying uncertainty propagated in mathematics-embedded ontologies. This approach is very general and can be applied to ontologies with any kind of mathematic relationship. In this thesis, a use case regarding the SWARMS ontology is used to illustrate the usefulness of the proposal.

4.3.1. Mathematic models in ontologies

One of the critical problems that ontologies face is to support mathematical expressions as some concepts defined in ontologies might be computed through mathematical models taking other concepts as inputs. For those indirect concepts, their uncertainties are subject to the uncertainty considerations of input concepts. To clearly define concepts that have computational relationships with other concepts, allowance of expressing the underlying mathematical model in ontologies is needed. Mathematical models can be represented as a set of rules encapsulated into ontologies by employing Semantic Web Rule Language (SWRL) math built-ins. Further, Sanches-Macian *et al.* [141] presented an extension to SWRL to enable advanced mathematical support in SWRL rules. By using this extended SWRL, complex mathematical relationships and formulae can be described and included in ontologies. Gangemi [142] proposed a novel idea to import MathML [143] and OpenMath [144], which are standard XML-based languages for mathematical knowledge, into ontology annotations so that descriptions of mathematical relationships can be enabled.

Focusing on the mathematic relationships embedded in ontologies, how to characterize the uncertainty of ontological concepts from a holistic view attracts little focus [145]. In some specific domains, such as engineering surveying and material analysis, acquisition of the statistics of random concepts is significant to obtain an overall view of the variation range. In this way, the usage of the historical database could be maximized. Besides, with regard to mathematical models encased in ontologies, none of the existing work put their focus on researching how uncertainty is propagated from input entities to output entity through those models. Therefore, in this thesis, a novel algorithm, called Stochastic Reduced Order Model (SROM) [146], is presented

to solve the uncertainty quantification for ontologies with mathematical considerations. Using this algorithm, the uncertainty of concepts can be viewed from a high level which refers to a holistic probability distribution. In addition, this algorithm introduces a lower computational cost. The propagated uncertainty, which is regarded as high-level information, is very useful for further use, such as filtering and inference.

4.3.2. Stochastic Reduced Order Model (SROM)

The principle of the Stochastic Reduced Order Model (SROM) method is presented in this section. Let us assume that $\mathbf{X}$ is an N -dimensional random variable ($N \geq 1$) if $\mathbf{X}$ is jointly described by N independent variables. Then, $\mathbf{X}$ can be represented as $[X_1, X_2, X_3, \dots, X_N]$. The statistics properties of $\mathbf{X}$ are assumed to be fully known with marginal distributions, moments of order q , and correlation matrix specified as [26]:

$$F_i(x) = P(X_i \leq x) \quad (1)$$

$$u_i(q) = E(X_i^q) \quad (2)$$

$$\mathbf{r} = E[\mathbf{X}\mathbf{X}^T] \quad (3)$$

where i varies from 1 to N .

The definition of SROMs

In essence, an SROM $\tilde{\mathbf{X}}$ is a simplified random element which approximates the statistical properties of the random variable $\mathbf{X}$. $\tilde{\mathbf{X}}$ is composed of two components: a finite set of samples $\{\tilde{x}^{(1)}, \dots, \tilde{x}^{(m)}\}$ and a corresponding set of probabilities $\mathbf{p} = \{p^{(1)}, \dots, p^{(m)}\}$. Any element $p^{(l)}$ ($1 \leq l \leq m$) of the probabilities set is required to meet two laws: $p^{(l)} \geq 0$ and $\sum_{l=1}^m p^{(l)} = 1$. The range of $\tilde{\mathbf{X}}$, also referred to as model size m , is determined under the consideration of accuracy and computational cost. A large model size m is more likely to accurately approximate the statistics of $\mathbf{X}$ but the computational load increases [146]. With fulfilling the aforementioned conditions, $\tilde{\mathbf{X}}$, represented as sample-probability pairs $(\tilde{x}^{(l)}, p^{(l)})$, $l=1, \dots, m$, is able to have similar statistics as $\mathbf{X}$. Similar to $\mathbf{X}$, the distributions and moments of $\tilde{\mathbf{X}}$ can be given as follows:

$$\tilde{F}_i(x) = P(\tilde{x}_i \leq x) = \sum_{l=1}^m p^{(l)} I(\tilde{x}_i^{(l)} \leq x) \quad (4)$$

$$\tilde{u}_i(q) = E(\tilde{x}_i^q) = \sum_{l=1}^m p^{(l)} (\tilde{x}_i^{(l)})^q \quad (5)$$

$$\tilde{r}_{ij} = E[\tilde{X}_i \tilde{X}_j] = \sum_{l=1}^m p^{(l)} \tilde{x}_i^{(l)} \tilde{x}_j^{(l)} \quad (6)$$

where $I(S)$ is an indicator function and it has two possible values: 1 when S is true and 0 when S is false. Different SROMs can be generated with different sample-probability pairs $(\tilde{x}^{(l)}, p^{(l)})$, $l=1, \dots, m$. Nevertheless, they differ from each other with regard to the degree of approximation to the real statistics of $\mathbf{X}$. How to achieve a trade-off between accuracy and

computation complexity and therefore produce an optimal candidate SROM are a key point and will be discussed in next section.

Building SROMs

As stressed in section 4.3.2.1, there could exist many different SROMs to approximate the statistics of $\mathbf{X}$. Construction of SROMs alternatives can be achieved by different algorithms. Grigoriu [146] introduced the usefulness of three algorithms including *Dependent thinning*, *Integer optimization*, and *Pattern classification* in SROMs construction. Furthermore, Warner *et al.* [147] proposed a new algorithm to construct SROMs which is stated to be improved considerably with respect to efficiency and accuracy. In this paper, the commonly used algorithm in building SROMs, which refers to pattern classification, is briefly outlined as follows:

Algorithm 1 Pattern classification

1. Collect a set of n finite independent samples $\{\delta_1, \dots, \delta_n\}$ from the random variable $\mathbf{X}$. The sample size n should be large enough to accurately characterize the statistics of $\mathbf{X}$.
2. The set $\{\delta_1, \dots, \delta_n\}$ becomes the raw source to generate SROMs. From $\{\delta_1, \dots, \delta_n\}$, randomly extract subsets $\{\tilde{x}^{(1)}(\gamma), \dots, \tilde{x}^{(m)}(\gamma)\}$, where $m \leq n$ and $1 \leq \gamma \leq C_n^m$. It could obtain C_n^m different kinds of SROM candidates.
3. Taking each trial subset $\{\tilde{x}^{(1)}(\gamma), \dots, \tilde{x}^{(m)}(\gamma)\}$ as generator seeds [148], the uncertain space of $\mathbf{X}$ could be divided into m partitions $\mathcal{L}_l(\gamma)$ ($1 \leq l \leq m$) centered at $\tilde{x}^{(l)}(\gamma)$. Each region $\mathcal{L}_l(\gamma)$ contains $n_{(l)}$ points, where $\sum_{l=1}^m n_{(l)} = n$.
4. Let $d^{(l)}$ denote the summation of Euclidean distances between $n_{(l)}$ points located in $\mathcal{L}_l(\gamma)$ to the corresponding center point $\tilde{x}^{(l)}(\gamma)$.
5. Accumulate $d^{(l)}$ ($1 \leq l \leq m$) to get the overall distance $d = \sum_{l=1}^m d^{(l)}$.
6. Select the subset $\{\tilde{x}^{(1)}, \dots, \tilde{x}^{(m)}\}$ with the minimum d as the most potential SROM $\tilde{\mathbf{X}}_{(opt)}$.
7. The probability for $\tilde{x}^{(l)}$ is determined by the law $p^{(l)} = n_{(l)}/n$ such that the summation of $p^{(l)}$ could be 1.

At this point, an SROM with a set of sample-probability pairs $\{(\tilde{x}^{(1)}, p^{(1)}), \dots, (\tilde{x}^{(m)}, p^{(m)})\}$ is generated by the *pattern classification* algorithm. Likewise, other SROMs can be constructed by using other algorithms, such as the aforementioned ones: *Dependent thinning* and *Integer optimization*.

With a set of SROMs candidates available, an optimal SROM should be found by imposing the condition that $\tilde{\mathbf{X}}$ should have a similar probability law as $\mathbf{X}$. Discrepancy between statistics of $\tilde{\mathbf{X}}$ and $\mathbf{X}$ is considered as the primary factor to evaluate the performance of different SROMs. The essential discipline is to select an SROM which minimizes the discrepancy from a set of alternative SROMs. The discrepancy is measured by an objective function with three components which express differences of the marginal distributions, the marginal moments, and the correlation matrices between $\tilde{\mathbf{X}}$ and $\mathbf{X}$, respectively. The definitions for the objective function are as follows:

$$e_1(\tilde{\mathbf{x}}, \mathbf{p}) = \sum_{i=1}^n \sum_{l=1}^m (\tilde{F}_i(\tilde{x}_i^{(l)}) - F_i(\tilde{x}_i^{(l)}))^2 \quad (7)$$

$$e_2(\tilde{\mathbf{x}}, \mathbf{p}) = \sum_{i=1}^n \sum_{q=1}^{\bar{q}} (\tilde{u}_i(q) - u_i(q))^2 \quad (8)$$

$$e_3(\tilde{\mathbf{x}}, \mathbf{p}) = \sum_{i,j=1,\dots,n; j>i} (\tilde{r}_{ij} - r_{ij})^2 \quad (9)$$

$$e(\tilde{\mathbf{x}}, \mathbf{p}) = \alpha_1 e_1(\tilde{\mathbf{x}}, \mathbf{p}) + \alpha_2 e_2(\tilde{\mathbf{x}}, \mathbf{p}) + \alpha_3 e_3(\tilde{\mathbf{x}}, \mathbf{p}) \quad (10)$$

where α_1, α_2 , and α_3 are weighting factors to define the relative importance of differences between marginal distributions, marginal moments, and correlations in the overall discrepancy between $\tilde{\mathbf{X}}$ and $\mathbf{X}$. Coefficients α_1, α_2 , and α_3 can be set with different constants in particular applications so as to ensure that each error component has the same importance or to emphasize the SROM's ability to represent a particular statistic of $\mathbf{X}$ over other statistics.

SROM-based uncertainty quantification

This section describes the procedures how the SROM algorithm is used to quantify the uncertainty propagation from multiple uncertain variables to the output with the deterministic solver known. The statistics of the output variable, which is jointly affected by multiple uncertain variables can be approximated with low computational cost and high accuracy by using the SROM-based method. The SROM-based uncertainty quantification method is outlined as follows:

Algorithm 2 The SROM-based Uncertainty Quantification

1. The mathematical relation between multiple input variables and the output variable Y is modelled by a deterministic solver which could be a set of mathematical relations. Multiple input variables can be merged to construct the random input variable $\mathbf{X}$ (described by $\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n$).
2. The statistics of the input variable $\mathbf{X}$ are known beforehand. The method introduced in section 4.3.2.2 is applied to generate its SROM $\tilde{\mathbf{X}}$, with sample-probability pairs $\{(\tilde{x}^{(1)}, p_x^{(1)}), (\tilde{x}^{(2)}, p_x^{(2)}), \dots, (\tilde{x}^{(m)}, p_x^{(m)})\}$, where $m \leq n$.
3. The SROM of the output variable $\tilde{y}^{(i)}$ ($1 \leq i \leq m$) is obtained through m deterministic calculations with $\tilde{x}^{(i)}$.
4. Sequentially, the probabilities of $\tilde{y}^{(i)}$, denoted as $p_y^{(i)}$, can be obtained using $p_y^{(i)} = p_x^{(i)}$ because the acquisition of $\tilde{y}^{(i)}$ is subject to the occurrence of $\tilde{x}^{(i)}$.
5. Then the SROM of the output variable $\tilde{\mathbf{Y}}$ is completely defined.

After obtaining the SROM-based output $(\tilde{y}^{(i)}, p_y^{(i)})$ ($1 \leq i \leq m$), the statistics of the output variable Y can be studied by the approximation result $\tilde{\mathbf{Y}}$. Therefore, the statistics of Y , estimated by $\tilde{\mathbf{Y}}$, can be measured by means of the marginal distributions, moments of order q and the standard deviation σ .

$$F(\theta) = P(\tilde{y} \leq \theta) = \sum_{i=1}^m p_y^{(i)} I(\tilde{y}^{(i)} \leq \theta) \quad (11)$$

$$E(\tilde{y}^q) = \sum_{i=1}^m p_y^{(i)} (\tilde{y}^{(i)})^q \quad (12)$$

$$\sigma(\tilde{y}) = \sum_{i=1}^m p_y^{(i)} (\tilde{y}^{(i)} - E(\tilde{y}^1))^2 \quad (13)$$

In a nutshell, the SROM-based method can accurately approximate the statistics of an uncertain output variable in presence of multiple input variables and in the meanwhile simplify the uncertainty quantification process with a lower computational load. It is also noted that the SROM solution is guaranteed to converge to the theoretical statistics of the output variable when the model size of input variable approaches infinity [149].

4.3.3. Application of SROM in mathematics-embedded ontologies

In this section, the first application of SROM to quantify uncertainty propagation in mathematics-embedded ontologies is presented. In addition, a specific case study is provided to show the usefulness of SROM in ontologies with regard to characterizing the uncertainty propagation in the next chapter.

Figure 47. SROM-based uncertainty propagation in ontologies.

Data obtained from specific domains are encapsulated in ontologies as instances under relevant entities. In general, data which are going to be constructed in ontologies are classified into two categorizations: direct and indirect [150]. The differentiation of direct and indirect information comes from the obtainment complexity. Indirect information is more complex to acquire and needs additional computation or inference *etc.* Therefore, a variety of entities in ontologies is obtained as outcomes of computational models which involve a set of direct entities as inputs. Thus, uncertainties existing in input entities will jointly affect the accuracy of the output entity through the computational model. The characterization of uncertainties becomes of significance in such kind of mathematical and computational models of complex processes and data. In this sense, the SROM algorithm can be employed to address this problem in ontologies from a statistical aspect. The process of SROM to propagate uncertainty in ontologies can be illustrated in Figure 47.

Let us assume that entity R is a concept representing an abstraction of a class of individuals with similar attributes in an ontology. R is determined as conclusions drawn from a mathematical model. The mathematical model, also referred to as a deterministic solver,

describes the relationship between the input entities and the output entity. Instances of R could be variable and random affected by the uncertainties of input entities. From this point, different entities defined in the ontology could be regarded as independent variables in the SROM-based consideration. The SROM algorithm can be used to construct an SROM which represents an approximation of statistics of multiple input variables. Instances of different input entities form the raw source to generate an SROM. The core principle is to integrate multiple inputs as a multidimensional variable. As a result, the uncertainties of different input concepts are modeled by the multidimensional variable. An optimal SROM of the input variable could be produced by using the method introduced in section 4.3.2. Afterward, following the steps introduced in section 4.3.2 this SROM which contains similar statistics of input entities is applied in the mathematical model to generate the SROM of output entity. Therefore, the statistical law of the output entity can be approximated and recovered by applying its SROM in equations (11-13). Based on the obtained statistics, the uncertainty of output entity (e.g., mean, standard deviation, and the variation range) can be extracted.

The application of SROM in quantifying uncertainty in ontologies can bring three benefits: 1) considerably reducing the computational cost, 2) accurately obtaining the statistics of the uncertain and random output entity, and 3) generally suiting ontological uncertainty propagation with any kind of mathematical models because of its non-intrusive feature. In the following section, a case study will be given to demonstrate the usefulness of the SROM algorithm in quantifying uncertainty propagation in mathematics-embedded ontologies.

4.3.4. A case study

In this section, the proposed SROM-based uncertainty quantification method will be shown through a case study. Assuming that underwater vehicles are dispatched to survey undersea cable on field, it is considered that a set of attributes, such as the radius of the cable, the length of the surveyed cable, and the volume of the surveyed cable, could be important criteria for studying the undersea cable (e.g., studying the electromagnetic compatibility characteristic of the cable in an undersea environment). Radius and length of the undersea cable can be measured by the underwater vehicles while its volume can be calculated based on these two parameters given the assumption that the shape of the cable is a standard cylinder. In this scenario, the undersea cable can be an instance of concept *OtherObject* defined in the environment recognition & sensing domain-specific ontology. Three new data properties, *hasRadius*, *hasLength*, and *hasVolume*, can be added to the SWARMS ontology in order to specify the relationship between

undersea cable and its attributes, radius, length, and volume. With these application extensions, it is able to represent this scenario by populating measured data into the SWARMS ontology by means of instances. In addition, the indirect attribute, volume, which is obtained based on radius and length, can be specified using the SWRL rule in the SWARMS ontology.

$$Volume = \pi * r^2 * l \quad (14)$$

The SWRL rule, which defines the mathematical relationship between radius, length and volume can be seen as follows:

OtherObject (?x), hasRadius (?x,?y), hasLength (?x,?z), swrlb:multiply (?r,?y,?y), swrlb:multiply (?h,?r,?z), swrlb:multiply (?v,?h,3.1415926)- >hasVolume (?x,?v)

Due to a variety of reasons, such as imperfect instruments and harsh condition, the radius and length measurements are associated with uncertainty, particularly referring to randomness and inaccuracy. Thus, the volume value, which is calculated from them, can be variable in a specific scale. It is necessary to make the uncertainty analysis on the volume of the inspected cable as its statistics, such as the variation range, could provide merits for further use, such as important inputs for studying its electromagnetic compatibility

In this case study, the studied variable volume is affected by two variables radius and length. The SROM-based method can construct a reduced model which consists of fewer samples from radius and length with the capability to represent the similar statistics of the original database of radius and length.

To deal with two uncertain input variables with the SROM method, the idea is to integrate these uncertainty sources into a bidimensional variable and then construct an SROM to globally approximate the overall uncertain input space. In this case, two variables *Length* and *Radius* are merged into a bivariate variable $X = [Length, Radius]$, where cardinality D equals 2. Instances of *Length* and *Radius* are filled into the input variable and treated as samples of X . Up to this point, the input variable X is completely defined. It is noted that in real cable surveying scenarios, a history of length and radius data can be acquired by instruments. Here, for the demonstration purpose, uncertain *Length* and *Radius* variables are assumed to have Gaussian distributions and a series of instances are randomly selected from the known distribution. However, it is worth noting that the SROM method can be applicable for any type of probability distribution.

More specifically, let us assume that class *Length* contains 10^4 instances and the dataset is randomly selected from a Gaussian distribution with mean $E(l) = 30 \text{ cm}$ and standard deviation

$\sigma(l) = 0.8 \text{ cm}$. Similarly, class *Radius* has a historical dataset which contains 10^4 instances randomly selected from a Gaussian distribution with mean $E(r) = 3.5 \text{ cm}$ and standard deviation $\sigma(h) = 0.01 \text{ cm}$. Then, the input variable X has 10^4 samples of length and radius values. As can be seen in Figure 48, each sample of X can be represented as a blue star marker in a plane which sets *Length* as the x-axis and *Radius* as the y-axis. The coordinates of the blue star marker show a pair of possible values for *Length* and *Radius* to calculate *Volume*.

Figure 48. The distribution of 10^4 instances of Length and Radius.

With the raw source of the input variable X , the SROM-based method could be used to extract a suitable SROM which contains a small amount of samples and is able to reflect the original distribution of X . By applying the algorithm introduced in section 4.3.2, a SROM $\tilde{X}$ can be produced to globally approximate the statistics of *Length* and *Radius*. For instance, the model size is set with a very small number: 20 samples to generate a SROM of uncertain inputs *Length* and *Radius*. Twenty green points, which are scattered in Figure 48 to ensure the exploration of the entire uncertain region X , are selected by applying the algorithm introduced in section 4.3.2. These twenty samples and their corresponding probabilities form the optimal SROM $\tilde{X}$. Each SROM sample is able to maximize the similarities between other samples which are located in

the same Voronoi region and itself. Thus, the produced SROM $\tilde{X}$ can be guaranteed to have similar statistics as X to a certain extent.

With different model size m , the approximation of statistics of *Length* and *Radius* could be different and therefore, the uncertainty quantification of output *Volume* could be different. To validate the performance of the SROM-based method, the result after 10^4 deterministic calculations is set as the benchmark. The benchmark Cumulative Distribution Function (CDF) [151] of *Volume* is drawn in a blue curve in Figure 49. The comparison of the SROM-based *Volume* output ($m = 20$) with the benchmark in terms of constructing *Volume* CDF is also presented in Figure 49.

Figure 49. CDF of *Volume* obtained by SROM-based method ($m=20$) and benchmark.

The SROM-based method, here with 20 samples, is able to approximate the statistics of *Volume* with a slight difference. As shown in Figure 49, the predicted CDF by the SROM method with model size 20 can reflect the general shape of the reference CDF. The computational cost is dramatically reduced by a factor of $10^4/20 = 500$. The model size 20 is reasonable as the approximated CDF matches the reference CDF in good agreement and the computational cost is kept low. When changing the model size, the difference between SROM-based results with

benchmark will change accordingly. Figure 50 depicts the SROM-based approximation of CDF of *Volume* with model size 40, 60, 80, and 100.

Figure 50. CDF of Volume obtained by SROM-based method and benchmark.

As can be seen from Figure 50, introducing more samples can steadily converge the reference CDF and get a better approximation. At the size of 60, the CDF of *Volume* can be very accurately recovered while it still requires less computational load than the benchmark. The rate of acceleration in computation is still notable which is $10^4/60 \approx 166.7$. When the model size reaches 100, the recovered CDF is almost identical as the benchmark result which shows that the SROM-based method has a very fast convergence rate to recover the benchmark result.

Figure 51. Error of mean of Volume.

In addition to providing the distribution information, the SROM method can also obtain a very accurate prediction of the mean and standard deviation of *Volume*. As Figure 51 shows, even at the model size of 5, the mean of *Volume* can be accurately approximated and the predicted mean of *Volume* is identical to the reference mean of *Volume* using 5 samples.

Figure 52. Error of Standard deviation of Volume.

With regard to the approximation to the reference standard deviation of *Volume*, the convergence rate is slower. As can be seen in Figure 52, increasing model size can steadily converge to the benchmark. Using the same amount of samples, the performance of predicting

standard deviation of *Volume* is not as ideal as predicting mean of *Volume*. But at the small size of 5, the error of standard deviation of *Volume* is still below 7%.

To show the performance of predicting mean and standard deviation of *Volume* using more samples, Table 4 presents different SROM-based results obtained by 20, 40, 60, 80 and 100 samples, accordingly.

Table 4. Comparisons between SROM-based results and reference

Attributes of Volume	SROM Model Size					REFERENCE
	20	40	60	80	100	10000
Mean	1154.6	1154.6	1154.6	1154.6	1154.6	1154.6
Standard deviation	30.5971	30.9048	31.0106	31.0596	31.0967	31.2506

As can be seen in Table 4, even with a very small model size of 20, the mean of *Volume* can be accurately obtained. The standard deviation by the SROM method is still accurate to a certain extent. The slight difference existing in the SROM-based results and the benchmark is affordable compared with the significant acceleration in computations. Taking the SROM-based result with model size 20 as an example, the SROM-based standard deviation is within the error of 5%.

Based on the mean and standard deviation of *Volume* obtained by the SROM method ($m = 20$), *Volume* can be predicted to likely vary within this interval: [1060.8482, 1248.3518]. This variation range predicted by the SROM-based method with model size 20 has very small difference compared with the benchmark which bounds the variation range as [1062.8087, 1246.3913]. With a bigger model size, the SROM-based prediction on statistics of the output variable can be refined and get a better convergence to the benchmark. It is worth noting that even with a small sized SROM, the uncertainty propagated from input entities to the output entity can be quantified with a reasonable degree of accuracy and a low computational cost. Since the increase of model size could guarantee a very good approximation to the reference result, a trade-off between accuracy and computational complexity should be figured out to suit different requirements of applications.

By using the SROM-based method, the uncertainty of the output entity is quantified from a statistical aspect so as to obtain the holistic variation range. This method can be used in the phase of ontology reasoning which aims to deduce more useful information from raw information. In this case study, instances of entities *Length* and *Radius* are low-level information. After applying the SROM-based method, the statistics of *Volume* are achieved as high-level information.

According to the previous case study, it has been proven that the SROM-based method can considerably reduce the computational cost while guaranteeing a good prediction of statistics of the output variable. The predicted uncertainty information of *Volume*, mainly referring to mean, standard deviation, and CDF, can be very useful to quantify the statistics of *Volume*. In addition, the statistics of *Volume* can be regarded as high-level information for other usages. For example, newly obtained instances of *Volume* could be removed as outliers under the consideration of its statistics. The change of *Volume* within a period of time, estimated by the variation range, can be considered as useful information to study the electromagnetic compatibility characteristic of the inspected undersea cable.

The SROM-based method is non-intrusive to the mathematical model which results in its compatibility and usability in all kinds of mathematical models regardless of different complexities. The selection of model size could be flexible, but it mainly should be based on consideration of computational cost and accuracy to fit in different needs of applications. The SROM-based method is demonstrated using a bivariate case study. This method could be also applicable to multiple variables. But the convergence rate could be different with regard to the number of input variables.

5. Context reasoning for the cooperation of underwater robots

5.1. The hybrid context reasoning mechanism

To correctly select reasoning techniques, two key factors are necessarily considered [12]. The performance of a specific reasoning technique is regarded as the primary criterion. In addition, given that different modeling methods are available, the reasoning techniques selected should be compatible and be supported by the specific modeling technique used for knowledge representation. Therefore, based on the analyses on existing context reasoning methods and the modeling technologies chosen for the SWARMS project, a hybrid context reasoning mechanism is needed and proposed in this thesis. The proposal is to loosely couple three different context reasoning, namely, ontological, rule-based, and MEBN reasoning techniques. It is believed that the combination of different context reasoning methods can enhance the overall performance by mitigating each method's weakness using others' strengths. As can be seen in Figure 53, the hybrid context reasoning functions provided by the context reasoner are mainly used by the data access manager component. The data access manager component is one component that is defined in the SWARMS middleware and it is responsible for controlling access to the ontology model and managing all requests regarding the ontology model, such as inserting data, modifying data, and querying data. Through data access manager, the reasoning capabilities provided by the context reasoner are applied to the ontology model, either to the deterministic part or probabilistic part. According to different reasoning requirements, the context reasoner can provide the most suitable reasoning method.

Figure 53. The hybrid context reasoning mechanism.

5.1.1. Ontological reasoning

Ontological reasoning, also referred to as description logic reasoning, uses a set of constructs, such as *TransitiveProperty*, *SymmetricProperty*, *AsymmetricProperty*, *ReflexiveProperty*, *IrreflexiveProperty*, *subClassOf*, *subPropertyOf*, *disjointWith*, *equivalentTo*, and *inverseOf*, to specify a terminological hierarchy and assert inner restrictions. It is mainly supported by two semantic web representation languages, RDF(S) and OWL. Tableau algorithm is the de facto standard reasoning algorithm that is employed in ontological reasoning. Ontological reasoning enables several kinds of operations, including concept satisfiability, consistency check, class subsumption, and logic inference. Therefore, by means of ontology-based reasoning, some facts that are implicit in the ontology given explicitly stated facts can be deduced. Table 5 presents a list of ontology-based reasoning rules [152].

Table 5. Partial ontological reasoning constructs

Ontological Reasoning Constructs	Specifications	Examples
subClassOf	Indicating that concepts are sub-concepts of a super class and are transitive.	$(?A \text{ rdfs:subClassOf } ?B) \wedge (?B \text{ rdfs:subClassOf } ?C) \Rightarrow (?A \text{ rdfs:subClassOf } ?C)$
subPropertyOf	Specifying that properties are sub-properties of a super property and are transitive.	$(?A \text{ rdfs:subPropertyOf } ?B) \wedge (?B \text{ rdfs:subPropertyOf } ?C) \Rightarrow (?A \text{ rdfs:subPropertyOf } ?C)$
TransitiveProperty	Specifying that the characteristic of properties is transitive.	$(?p \text{ rdf:type owl:TransitiveProperty}) \wedge (?A ?p ?B) \wedge (?B ?p ?C) \Rightarrow (?A ?p ?C)$
SymmetricProperty	Specifying that properties are symmetric.	$(?p \text{ rdf:type owl:SymmetricProperty}) \wedge (?A ?p ?B) \Rightarrow (?B ?p ?A)$
DisjointWith	A member of one class cannot simultaneously be an individual of other classes that are tagged as disjointWith.	$(?A \text{ owl:differentFrom } ?B) \wedge (?C \text{ rdf:type } ?A) \wedge (?D \text{ rdf:type } ?B) \Rightarrow (?C \text{ owl:differentFrom } ?D)$
inverseOf	Indicating that knowing one of two properties allow us to imply the other.	$(?p \text{ owl:inverseOf } ?r) \wedge (?A ?p ?B) \Rightarrow (?B ?r ?A)$

As all context data are formalized in the SWARMS ontology, it is natural to apply ontology-based reasoning over ontology classes and instances. Several reasoners, including Pellet reasoner, HermiT [153] reasoner, and FaCT++ [154] reasoner, have been developed to support the ontological reasoning in ontologies. However, ontology-based reasoning lacks the ability to express complex relations or rules for reasoning tasks. In addition, it is unable to deal with missing values and reason over uncertainties.

5.1.2. Rule-based reasoning

To compensate for the ontological reasoning's inability to represent complex rules and relations, SWRL is adopted to define rules in the proposed hybrid context reasoning mechanism. SWRL represents a combination of OWL and RuleML [155]. The expressivity of OWL is extended by including Horn-like [156] rules enabled by SWRL. SWRL rules contain unary predicates for the description of classes and data types, binary predicates for expressing object properties, and several built-in n-ary predicates, such as math built-ins, comparisons built-ins, and string built-ins. In essence, SWRL rules are defined in an antecedent-and-consequent, also referred to as if-then, implication. If the set of conditions specified in the antecedent part is met, then the SWRL rule is executed and the assumptions specified in the consequent part can be inferred as high-level information. A typical SWRL rule can be of the following form:

$$a_1 \wedge a_2 \wedge \dots \wedge a_n \rightarrow b_1 \wedge b_2 \wedge \dots \wedge b_m$$

Where a_i and b_i are OWL atoms of the following forms:

- Concepts, e.g., $C(x)$, where C is an OWL description, and x is either a variable, an OWL individual or a data value.
- Object properties, e.g., $R(x,y)$, where R is an object property and x, y are either variables, individuals or data values.
- Data properties, e.g., $D(x,y)$, where D is a data property, x is a variable or individual while y is a data value.
- $B(x_1, x_2, \dots)$, where B is a built-in relation (e.g., math built-in) and $x_1, x_2, \dots$ are either variables, individuals or data values.
- $\text{SameAs}(x,y)$ or $\text{differentFrom}(x,y)$ where x, y are either variables, individuals or data values.

Due to the aforementioned SWRL features, marine experts can encode their knowledge and experience into SWRL rules. Rules creator can insert them into the ontology model. A simple SWRL rule, that has been defined in the SWARMs ontology, can be seen as follows:

SWRL rule-based reasoning: *Robot(?x), hasBatteryLevel (?x,?y), swrlb:greaterThan(?y, 60%) $\rightarrow$ RobotCandidateForMission(?x).*

The rule above is very straightforward and expresses an assumption that a robot can be regarded as a potential candidate to carry out a task if its battery level is higher than 60%. Once

the ontology model is populated with instances, the rule-based reasoning can infer over those instances based on the pre-defined rule. Afterward, more useful information (e.g., robot_Alister is a candidate to execute the task) can be derived based on low-level contexts (e.g., robot_Alister is an underwater robot and its current battery level is 80%).

With the allowance of encasing user-defined rules in the ontology model enabled by SWRL, the reasoning task can be augmented in terms of logicity and human readability. Pre-insertion and formalism of SWRL rules are supported by the Protégé. The rules creator component also intends to provide interfaces for operators to define and insert rules during missions. Reasoners Pellet and HermiT can enable automatic reasoning based on SWRL rules. Despite the fact that rule-based reasoning has several useful features (e.g., ease of use, ease of implementation, extendibility, and human readability), it also has several limitations remaining to be solved. Manually encasing a large volume of user-defined rules in the ontology model might lead to error-proneness and undecidability in the reasoning phase. Besides, in terms of the ability to define changing events, they lack interoperability and reusability. Similar to ontological reasoning, rule-based reasoning is unable to reason over uncertainties.

5.1.3. MEBN reasoning

As introduced in section 4.2.6, the SWARMs ontology adopts the PR-OWL ontology to represent context uncertainty based on the MEBN theory. In this way, the SWARMs ontology provides substantial support for using the MEBN reasoning over its probabilistic annotations. The principles of MEBN have been presented in section 2.1.2. In essence, the MEBN reasoning is based on Bayesian reasoning. The major advantage of the MEBN reasoning over the BN reasoning is that it can represent repeated structures in a BN in a compact way. Random variables in an MEBN model can be dynamically instantiated and form Situation Specific Bayesian Networks (SSBNs) to represent different scenarios. The generation of SSBN from an MEBN model follows Laskey's algorithm. More details on how to instantiate an MEBN to an SSBN can be found in [68].

The SSBN can make inferences under uncertainties based on standard BN reasoning. As highlighted previously, the UnBBayes tool is a powerful tool that leverages the capability of creating MEBN models and facilitating MEBN reasoning. The UnBBayes tool provides both Graphical User Interface (GUI) and Java Application Programming Interfaces (APIs) to build MEBN models, generate probabilistic ontologies to represent the MEBN models, and make

uncertainty reasoning. To the best of my knowledge, the UnBBayes tool is the only tool that supports MEBN reasoning available in current research.

The UnBBayes GUI makes it possible to model the different MFragS that will form an MTheory, as if they were BNs, but with specific nodes for OV_s, CN_s, and RN_s. Resident Nodes Conditional Probability Tables (CPT_s) can be edited and a wide set of operations can be used for defining the distributions. Context Nodes also offer a wide variety of conditional structures and operators to model the constraint satisfaction functions. The GUI also offers a querying mode and the options to manually insert new entities, entity instances, and findings of those entities or about any RV. However, they are insufficient for the purposes of providing the MEBN reasoning in the context reasoner component, as there is a need to make use of that functions in a programmatically way. The API helps on this aspect by offering some facade classes that allow the user to execute a set of common operations to interact with the MEBNs defined within a UnBBayes PR-OWL file and a knowledge base. Particularly, a specific class, `TextModeRunner.java`, provided by the UnBBayes framework, implements a lot of functionalities, such as loading MEBN models and knowledge bases and executing Laskey's algorithm in order to execute the MEBN reasoning. This class can be used by the implementation of the MEBN reasoning API in the context reasoner component.

Compared with the ontological and rule-based reasoning, the MEBN reasoning is more application specific. In other words, the MEBN reasoning is more oriented to solve specific application/scenario problems instead of reasoning on the whole SWARMs ontology. In this sense, the MEBN reasoning will reason about problems which contain uncertainty considerations. When dealing with specific uncertainty problems, MEBN models can be created to represent those problems and make probabilistic reasoning.

5.2.A preliminary proof of concept

In this section, a preliminary proof of concept, namely how the three context reasoning methods are envisaged to be useful in the SWARMs project, is provided. Specifically, the ontological and rule-based reasoning results are obtained by using Protégé and its Pellet reasoner plug-in. In addition, the scenario described in section 4.2.6, namely the oil spill monitoring scenario, is revisited to demonstrate the usefulness of the MEBN reasoning. The UnBBayes GUI is used to facilitate the MEBN reasoning about this scenario.

5.2.1. Ontological reasoning results

The Pellet reasoner, as a plug-in in Protégé, can provide both ontological and rule-based reasoning capabilities. By opening the SWARMs OWL file and selecting the Pellet reasoner in Protégé (shown in Figure 54), the SWARMs ontology is found to be consistent.

Figure 54. Applying the Pellet reasoner to facilitate ontological reasoning.

As emphasized in section 5.1.1, ontological reasoning can make a set of basic but essential inference, such as class subsumption and checking inconsistency. In order to better show what ontological reasoning actually could do, two detailed ontological examples are presented in the following.

Example 1 (Inconsistency problem): An instance, *test1*, is inserted in the Protégé as an individual of both class *Seabed* and *Sensor*. However, the class *Seabed* and *Sensor* are defined to be disjoint with each other. After repeating the same process aforementioned, the reasoning result can be seen in Figure 55. The ontology elements, that are detected as inconsistent, are highlighted in red. It shows the capability of the ontological reasoning in checking inconsistency of ontologies.

Figure 55. An example of ontology inconsistency in the SWARMs ontology.

Example 2 (Transitive property): The *biggerThan* object property is defined as *transitive* in the SWARMs ontology and its domain and range are Entity. After Inserting three new instances, *test2*, *test3*, and *test4*, as individuals of class Entity and specifying that *test2 biggerThan test3* and *test3 biggerThan test4*, a new piece of information, $(?biggerThan \text{rdf:type } owl:TransitiveProperty) \wedge (?test2 ?biggerThan ?test3) \wedge (?test3 ?biggerThan ?test4) \Rightarrow (?test2 ?biggerThan ?test4)$, can be inferred by applying the ontological reasoning. As can be seen in Figure 56, a new piece of information, *test2 is bigger than test4*, has been inferred based on the ontological reasoning.

Figure 56. The newly inferred knowledge shown in Protégé.

5.2.2. Rule-based reasoning results

Recalling the rule example (if the battery level of a robot is higher than 60%, then it can be a candidate for planning a mission) presented in section 5.1.2, this rule can be written into the SWARMs ontology through the SWRLTab (shown in Figure 57) embedded in Protégé.

Figure 57. The RobotCandidateForMission rule in Protégé.

To show the rule-based reasoning, a new instance, `robot_Alister`, is created and inserted into the SWARMs ontology as an individual of class `Robot`. The battery level of `robot_Alister` is 80%. After applying the rule-based reasoning provided by the Pellet reasoner, a new piece of information can be obtained. As shown in Figure 58, the reasoning result indicates that `robot_Alister` is also an instance of class `RobotCandidateForMission` since its battery level is higher than 60% which meets the antecedent of the SWRL rule.

Figure 58. The rule-based reasoning result in Protégé.

The RobotCandidateForMission rule is just a simple, but straightforward, example for showing the capability of the rule-based reasoning. Likewise, more rules can be created through MMT and translated by rules creator. And by taking those rules into account, the rule-based reasoning can obtain more useful information.

5.2.3. MEBN reasoning results

In order to make reference easier, the description of the oil spill monitoring scenario provided in section 4.2.6 is repeated here. The proactive detection of oil spills is an important means to minimize the damage of spills to the marine environment. Assuming a set of context can be obtained by a set of SWARMs vehicles and based on this contextual information they can collaboratively detect the occurrence of oil spills. After detecting oil spills in a specific marine region, it is significant to predict the severity level of the spilled region so that remedial measures (e.g., clean-up or containment) could be taken accordingly. The estimation of the severity level of the specific area depends on available raw context information, including the thickness of spills, the estimated size of spills, the weather condition, and the currents. In addition, the severity level of the specific area could be influenced by different spills concurrently occurred in the same area.

The MEBN model, described in section 4.2.6, can represent the scenario and encode the joint probability distribution of the severity level. The MEBN model, shown in Figure 60, can be instantiated with different possibilities. In other words, its MFragments and nodes can be collectively grouped to form an SSBN to represent the same scenario with different findings. For instance, if two oil spills (spill_1 and spill_2) are detected in a specific region of the ocean (region_1). Spill_1 is observed as being thick and large and spill_2 is detected as being thin and small. In addition, the weather condition above the spilled marine region is known to be inclement. Currents in the region_1 fluctuate very strongly.

In this example, the obtained findings are listed in Figure 59. As shown in Figure 59, all necessary context information for estimating the severity level of the spilled region is obtained.

Findings:

```
isA(spill_1, Spill)=True
isA(spill_2, Spill)=True
isA(region_1, Region)=True
Location(spill_1)=region_1
Location(spill_2)=region_1
Weather(region_1)=Inclement
Currents(region_1)=Strong
EstimatedSize(spill_1)=Large
EstimatedSize(spill_2)=Small
Thickness(spill_1)=Thick
Thickness(spill_2)=Thin
```

Query:

```
SeverityLevel(region_1)?
```

Figure 59. Findings in this scenario.

The UnBBayes GUI can read and open the MEBN model, namely, the probabilistic ontology file, created in section 4.2.6. The findings can be manually populated into the MEBN model and stored as an independent file (also referred to as knowledge base). The knowledge base system used by the UnBBayes project to handle findings is “Powerloom KB” [157] which is capable to programmatically update and store the findings in a .plm format. The declarations for findings are in a declarative LISP syntax. For instance, the evidence that currents in region_1 are strong can be created through GUI as shown in Figure 61.

Figure 60. The MEBN theory shown in the UnBBayes GUI.

Figure 61. Creation of an instance of Currents.

With the complete findings specified in Figure 59, the MEBN can be instantiated as an SSBN after receiving a query (in this case, the SeverityLevel node and its argument, region_1, are the query target). The SSBN, that represents the scenario and is able to answer the query, can be seen in Figure 62.

Figure 62. SSBN generated for the scenario with complete findings.

Based on standard BN inferences, the SSBN can update beliefs of different random variables, such as the region_1 with the existence of two spills could be estimated as very serious with a probability of 88% and there is a 90% chance that the spread speed of spill_1 and spill_2 is fast. This high-level context could be sent to the MMT for further exploitation. For instance, it could be used by operators as a parameter to conceive a plan for underwater robots so that they can take remedial measures accordingly. It is also worth noting that the MEBN reasoning can reason under incomplete and uncertain context. For instance, in the same scenario, assuming that the thickness of two spills is unknown, the nodes in the MEBN model can be collectively instantiated as an SSBN to represent the specific scenario. The newly generated SSBN with incomplete findings can be seen in Figure 63. As shown in Figure 63, the instantiated nodes of Thickness, namely, Thickness_spill_1 and Thickness_spill_2, are not populated with evidence.

But with unknown context, the SSBN can also make inferences under uncertainties. The severity level of the spilled region can be inferred as high with a probability of 78.24%. It is a fact that this estimation is less reliable than the previous inference with complete findings known. However, it showcases that the MEBN reasoning can cater to real-world situations which are partially observable and it can deal with uncertainty reasoning given unknown data.

Figure 63. SSBN generated for the scenario with incomplete findings.

In this scenario, only two spill instances are exemplified as influence factors for the severity level. It is worth noting that the built MEBN could be flexible to accommodate to reasoning which involves more spill instances due to its modularity. Therefore, the same MEBN could be instantiated to deal with a varying number of entities.

6. Implementation and validation

This chapter will show the implementation of the proposed SWARMS ontology and the proposed hybrid context reasoning mechanism. In addition, the different context reasoning capabilities provided by the context reasoner will be validated through some examples.

6.1. Implementation scope and purpose

The implementation scope is limited to two context-aware framework components, ontology model and context reasoner. As can be seen in Figure 64, they are highlighted in purple. Specifically, the proposed SWARMS ontology will be deployed in the ontology model component. In addition, the proposed hybrid context reasoning mechanism will be implemented for the context reasoner component. The ontology model component and context reasoner component will be developed in the Java language. Particularly, the main focus is to implement a set of Java APIs for the context reasoner component in order to provide different context reasoning capabilities. The purpose of the context reasoner is not to implement context reasoning algorithms from scratch, but to integrate a set of existing context reasoners, including the Jena OWL reasoner [158], Pellet reasoner, and UnBBayes-MEBN reasoner, in order to provide a set of APIs that can be easily used by the data access manager to apply different context reasoning on the SWARMS ontology and get the reasoning results. The implementation of context reasoner highly takes advantage of existing reasoners for providing different context reasoning and integrates them in the context reasoner component.

Figure 64. Implementation scope within the semantic middleware.

6.2. The implementation of the proposed SWARMS ontology

As introduced in section 4.2.4, the Protégé tool is used to develop the proposed SWARMS ontology. The SWARMS ontology metrics can be seen in Figure 65.

Ontology metrics:	
Metrics	
Axiom	1039
Logical axiom count	646
Declaration axioms count	294
Class count	102
Object property count	104
Data property count	74
Individual count	14
DL expressivity	SRIQ(D)

Figure 65. The SWARMS ontology metrics.

As can be seen in Figure 65, there are 102 classes in total defined in the SWARMS ontology. In addition, the SWARMS ontology contains 104 object properties and 74 data properties. Currently, 14 individuals have been populated in the SWARMS ontology.

The OWL/XML syntax is chosen to save the implemented SWARMS ontology. An excerpt of the SWARMS ontology in the OWL/XML syntax is presented in the following. The full SWARMS ontology OWL file can be accessible in this web link (<http://www.swarms.eu/dissemination.html#publications>).

// An excerpt of the SWARMS ontology in the OWL/XML syntax

```
<Annotation>
<AnnotationProperty abbreviatedIRI="rdfs:comment"/>
<Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-
ns#PlainLiteral">
The Robotic Vehicles ontology models the robots and vehicles that are used in
the different SWARMS missions (then only the water world). Some important
elements that capture the model are: • Mobile robots are vehicles and robots
(polymorphism); • Robots are either autonomous robots, automated robots or
remotely piloted robots (disjunction); • Vehicles are either underwater
vehicles or surface vehicles (disjunction); • Vehicles are either motorized
or propelled by the environment / unmotorized (disjunction).
</Literal>
</Annotation>
<Annotation>
<AnnotationProperty abbreviatedIRI="rdfs:comment"/>
<Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-
ns#PlainLiteral">
```


This ontology model is defined targeting to characterize the environment and context data sensing. The environment is defined through a set of concepts, which are specified by particular properties, that define the surroundings of the location where a mission or task takes place involving robotic vehicles, e.g. AUV, ROV, USV. The environment domain models the relevant information that is addressed in the context awareness framework of SWARMS semantic middleware.

```

</Literal>
</Annotation>
<Declaration>
<DataProperty IRI="/vehicleNominalSpeed"/>
</Declaration>
<Declaration>
<DataProperty IRI="/entitySize"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/colocated"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/heavierThan"/>
</Declaration>
<Declaration>
<Class IRI="/Sensor"/>
</Declaration>
<Declaration>
<DataProperty IRI="/seabedPorosity"/>
</Declaration>
<Declaration>
<Class IRI="/Task"/>
</Declaration>
<Declaration>
<Class IRI="/Buoy"/>
</Declaration>
<Declaration>
<Class IRI="/WaterColumn"/>
</Declaration>
<Declaration>
<DataProperty IRI="/imageWidth"/>
</Declaration>
<Declaration>
<DataProperty IRI="/imageCoding"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/aimsFor"/>
</Declaration>
<Declaration>
<Class IRI="/ChangeEvent"/>
</Declaration>
<Declaration>
<Class IRI="/MotorizedVehicle"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/performedBy"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/hasHigherResolutionThan"/>
</Declaration>

```

```

<Declaration>
<ObjectProperty IRI="/hasLandmark"/>
</Declaration>
<Declaration>
<Class IRI="/MissionDuration"/>
</Declaration>
<Declaration>
<DataProperty IRI="/imageSize"/>
</Declaration>
<Declaration>
<DataProperty IRI="/sensorResolution"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/sensedAfter"/>
</Declaration>
<Declaration>
<Class IRI="/Mission"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/isDescriptionOf"/>
</Declaration>
<Declaration>
<DataProperty IRI="/imageHeight"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/canReplace"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/allocatedTo"/>
</Declaration>
<Declaration>
<Class IRI="/Radio"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/hasWave"/>
</Declaration>
<Declaration>
<Class IRI="/Acoustic"/>
</Declaration>
<Declaration>
<NamedIndividual IRI="/pacific_water_surface"/>
</Declaration>
<Declaration>
<Class IRI="/UnderwaterRobot"/>
</Declaration>
<Declaration>
<Class IRI="/AcousticModem"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/isConstraintOf"/>
</Declaration>
<Declaration>
<Class IRI="/ManmadeEntity"/>
</Declaration>
<Declaration>
<DataProperty IRI="/pressure"/>
</Declaration>

```

```

<Declaration>
<ObjectProperty IRI="/taskObjectProperties"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/hasEntity"/>
</Declaration>
<Declaration>
<Class IRI="/AutonomousRobot"/>
</Declaration>
<Declaration>
<NamedIndividual IRI="/temp1"/>
</Declaration>
<Declaration>
<Class IRI="/VehiclePlan"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/isAchievedBy"/>
</Declaration>
<Declaration>
<Class IRI="/GPSPosition"/>
</Declaration>
<Declaration>
<DataProperty IRI="/zcoordinate"/>
</Declaration>
<Declaration>
<Class IRI="/VehicleCapability"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/source"/>
</Declaration>
<Declaration>
<DataProperty IRI="/gpsAltitude"/>
</Declaration>
<Declaration>
<Class IRI="/MissionConstraint"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/canProcess"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/belongsToMissionPlan"/>
</Declaration>
<Declaration>
<ObjectProperty IRI="/hasPressure"/>
</Declaration>
<Declaration>
<Class IRI="/ASV"/>
</Declaration>

```

The developed SWARMS deterministic ontology will be the knowledge base for testing the ontological and rule-based reasoning capabilities provided by the context reasoner component. In addition, the probabilistic ontology defined in section 4.2.6 will be used to verify the MEBN reasoning capabilities provided by the context reasoner. The Jena Ontology API [159] is chosen to manipulate the SWARMS ontology (including the deterministic part and probabilistic part) and

implement the ontology model component. Jena is a programming toolkit that uses the Java programming language. It provides a set of Application Programming Interfaces (APIs) [160] to manipulate ontologies, such as creating ontology models, loading ontologies from files or URI (Uniform Resource Identifier) [161], inserting ontology elements, removing ontology elements, consulting data by making SPARQL (SPARQL Protocol and RDF Query Language) [162] queries, and applying the ontological reasoning.

The process to load the SWARMS ontology using Jena API is as follows:

```
String root = "swarmsontology.owl";
Model m=ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM);

File f=new File(root);
try {
    m.read(new FileInputStream(f),"");
}
catch (FileNotFoundException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

It uses the ModelFactory methods provided by Jena for creating a standard kind of model and loading the SWARMS ontology from a local place. Therefore, the SWARMS deterministic ontology which is loaded into the model m can be used as the knowledge base to apply the ontological and rule-based reasoning.

6.3.The implementation of the context reasoner component

Different existing context reasoners will be employed and integrated into the context reasoner component. The procedures to apply ontological, rule-based, and MEBN reasoning are different and they will be specified in section 6.3.1. Section 6.3.2 will present the specific interfaces which provide different context reasoning functions in the context reasoner.

6.3.1. Specifications for the use of different context reasoning

In this section, the use of different context reasoning is specified. The process to invoke different context reasoning capabilities embedded in the context reasoner is illustrated using sequence diagrams.

Use of the ontological reasoning

The process to use the ontological reasoning can be seen in Figure 66. Whenever the SWARMs ontology in the ontology model component is manipulated (e.g., data access manager requires to insert new data into the SWARMs ontology, data access manager asks the ontology model component to delete/remove data), the ontological reasoning must be invoked by the data access manager in order to check the updated ontology and make the ontological reasoning. The reasoning results can be stored in the ontology model so that operators or vehicles can make queries to get newly inferred data.

Figure 66. The use of the ontological reasoning.

Use of the rule-based reasoning

Generally speaking, the rule-based reasoning can be used in two cases which are illustrated in Figure 67 and Figure 68, respectively. The first one is shown in Figure 67. Operators or marine experts can insert rules through the rules creator to regulate the data stored in the ontology model component. The rules creator could allow the insertion of specific SWRL rules. Users without ontological knowledge can specify rules using an if-then implication and the rules creator can translate them into corresponding SWRL rules. Once the rules are inserted into the ontology model component by the data access manager, the rule-based reasoning is invoked and applied on the ontology model taking rules into account. New facts can be inferred based on rules, be stored in the ontology model, and finally be consulted by MMT or vehicles. The other case is similar to the use of ontological reasoning. Whenever the ontology model is manipulated, the rule-based reasoning should be executed in order to make the rule-based reasoning based on the updated ontology.

Figure 67. The first kind of use of the rule-based reasoning.

Figure 68. The second kind of use of the rule-based reasoning.

Use of the MEBN reasoning

Figure 69 shows the process to use the MEBN reasoning. Specifically, when dealing with a specific scenario where context uncertainties are involved in, operators or marine experts can pre-define a probabilistic ontology based on the MEBN theories. The probabilistic ontology, also referred to as the MEBN model, can be created using the UnBBayes GUI tool and stored in a Universal Binary Format (UBF) [163] or OWL file. Therefore, this probabilistic ontology, as application-specific extensions, is part of the SWARMs ontology and it can be stored separately from the SWARMs deterministic ontology, in the ontology model component. The reasons of separating the SWARMs deterministic ontology and probabilistic ontology are two-fold. The first reason is that probabilistic ontologies are oriented to model specific applications/scenarios rather than providing a general representation as the SWARMs deterministic ontology does. In this sense, the probabilistic ontologies can be invalid or useless when shifting to new scenarios. Keeping probabilistic ontologies in a separate OWL or UBF file can guarantee an easier management, such as deleting useless probabilistic ontologies or adding new probabilistic ontologies. The other reasoning is that the UnBBayes APIs, that are adopted to programmatically provide MEBN reasoning, cannot reason over an OWL file which merges deterministic and

probabilistic ontologies. Whenever the ontology model, particularly referring to the probabilistic part, is manipulated (e.g., insert or remove data), the data access manager will update the knowledge base (containing findings or evidence) for the probabilistic reasoning. Afterward, the data access manager will invoke the MEBN reasoning from the context reasoner. The context reasoner can apply the MEBN reasoning on the probabilistic ontology in the ontology model component. Afterward, the data access manager can answer to queries regarding the MEBN model based on the reasoning results, such as the probability of a specific resident node/variable.

Figure 69. The use of the MEBN reasoning.

6.3.2. Implementation of different context reasoning capabilities in the context reasoner

The main purpose of the implementation of the context reasoner component is not to implement the three different context reasoning algorithms from scratch or improve their performance to a certain degree. Instead, it aims to maximize the use of reasoning capabilities provided by existing context reasoners and provide simple interfaces to access them. Specifically, the adopted existing context reasoners include the Jena OWL reasoner, the Pellet reasoner, and the UnBBayes-MEBN reasoner.

The component diagram for the context reasoner, data access manager, and ontology model can be seen in Figure 70. Three kinds of interfaces provided by the context reasoner component can be used by the data access manager in order to apply different context reasoning to the ontology model. These interfaces will be used for the data access manager or semantic query module to get more valuable information.

Figure 70. Component diagrams.

Specifically, the context reasoner component provides **ontologicalReasoning**, **ruleBasedReasoning**, and **mebnReasoning** interfaces. The descriptions of **ontologicalReasoning** and **ruleBasedReasoning** interfaces can be seen in Table 6 and Table 7, respectively.

Table 6. **ontologicalReasoning** interface

<i>(public)</i> <code>InfModel applyOntologicalReasoning(Model m)</code>	It will use the Jena OWL reasoner to provide the ontological reasoning on the specified model. In addition, the inferred model will be returned.
<i>(public)</i> <code>void updateOntologyWithOntologyInferredData(InfModel im, Model m)</code>	It updates the SWARMs deterministic ontology with the ontological reasoning results.

Table 7. **ruleBasedReasoning** interface

<i>(public)</i> <code>InfModel applyRuleBasedReasoning(Model m)</code>	It uses the Pellet reasoner to provide rule-based reasoning on the specified model. In addition, the inferred model will be returned.
--	---

<i>(public)</i> void updateOntologyWithRuleInferredData (InfModel im, Model m)	It updates the SWARMS deterministic ontology with the rule-based reasoning results.
--	---

It is worth noting that integrating ontologies and SWRL rules are a homogeneous approach because both ontologies and rules are embedded in a common logical language (e.g., OWL/XML is the chosen language in this dissertation). In such an approach, some ontology concepts and properties may be defined through rules. When intending to make the ontological and rule-based reasoning, the whole deterministic part of the SWARMS ontology needs to be reasoned. The major difference between the MEBN reasoning and the other two reasoning methods is that the MEBN reasoner only needs to be applied to the probabilistic part of the SWARMS ontology, namely the pre-defined MEBN model, and execute the Laskey's algorithm in order to reason about probabilities of variables in the MEBN model. It is assumed that data access manager can provide a set of methods in order to update the probabilistic knowledge base with newly obtained data. With the updated knowledge base, methods, such as fillFindings(MEBN, KnowledgeBase) and executeMEBNQuery (String node, String arg), can be adopted to update the knowledge base to the MEBN model and execute the Laskey's algorithm for reasoning on the instantiated MEBN model, namely, an SSBN model. The specific mebnReasoning interface defined in the context reasoner component can be seen in Table 8.

More details regarding the reasoning interfaces and description of the classes that implement those interfaces can be found in Appendix B.

Table 8. mebnReasoning interface

<i>(public)</i> void applyMEBNReasoning (String node, String argument)	It calls the Laskey's algorithm on the given MEBN model and uses the given findings. Based on the specified node and argument of the node, an SSBN is generated and the MEBN reasoning is applied in order to reason about the probabilities of the node's states.
<i>(public)</i> ProbabilisticNode getMEBNReasoningResult()	It returns the reasoned result, namely the queried random node. The returned probabilistic node can be further used by the data access manager to get the reasoning results, such as the probability distribution of the node.

6.4. Validation of the hybrid context reasoning mechanism

As described in the previous section, the methods provided by the context reasoner component will be used mainly by the data access manager. Due to the fact that the data access manager is still under development and it is also out of the scope of this thesis, the validation of the context reasoner is directly carried out using the SWARMS ontology in-memory instead of being done through the data access manager.

Different context reasoning capabilities provided by the context reasoner will be shown by using some examples. A preliminary quantitative analysis on the performance of different context reasoning will be given. Specifically, in order to validate the practical performance of the context reasoner, comparisons of reasoning time spent on inferring the SWARMS ontology with different amount of instances are provided. Tests are carried out using an Intel (R) Core (TM) i7-6498DU CPU @2.50 GHz 2.60 GHz processor equipped with a RAM memory of 8 GB in a machine operating under the 64-bit Windows 10 operating system.

6.4.1. Test on ontological reasoning

The deterministic part of the SWARMS ontology is taken as the knowledge base to test the performance of the ontological reasoning. As introduced in section 6.2, the SWARMS deterministic ontology contains 102 classes, 104 object properties, 74 data properties, 14 individuals and 1 SWRL rule in its current version.

After invoking the `applyOntologicalReasoning` function provided by the context reasoner component, the ontological reasoning is applied to the SWARMS ontology. The reasoning result, depicted in Figure 71, shows that the ontological reasoning is successfully applied to the SWARMS ontology and there is no inconsistency detected which is coherent to the reasoning result (introduced in section 5.2.1) obtained by the Protégé Pellet plug-in.

```
<terminated> ontologyReasoningValidation [Java Application] C:\Program Files\Java\jdk1.8.0_131\bin\javaw.exe (Jun 13, 2017, 10:13:15 AM)
The ontological reasoning on the deterministic part of the SWARMS ontology took 84 milliseconds.
The information inferred by the Jena OWL reasoner has been stored in the SWARMS ontology.
```

Figure 71. Ontological reasoning test.

Figure 71 shows the messages printed in the console after applying the ontological reasoning to the SWARMs deterministic ontology. As can be seen in Figure 71, the time for executing the ontological reasoning in the SWARMs deterministic ontology is 84 milliseconds. In addition, by using the `updateOntologyWithOntologyInferredData` method, the newly inferred facts by the ontological reasoning are stored in the SWARMs ontology.

Different SPARQL queries can be made through the semantic query component to the updated SWARMs ontology in order to get the newest information.

In real usages, the TBox (Terminology Box) [39], namely the vocabulary, including concepts and roles, of the SWARMs deterministic ontology will not be changed dramatically in terms of its size. It will only slightly be extended with application-specific terminologies. On the contrary, the ABox (Assertion Box) [39], namely assertions of individuals of either concepts or roles, will increase as amounts of data obtained from the real applications will be inserted. In other words, the SWARMs deterministic ontology will be populated with more individuals for concepts and relationships when being used in real scenarios. The total time required by applying the ontological reasoning on the SWARMs ontology is worth to be studied and quantified. Several indicative performance tests are done in order to examine the ontological reasoning performance. Specifically, the ontological reasoning is applied to the SWARMs ontology with a different number of individuals, such as 20, 40, 60, 80, 100, 120, 140, 160, 180, and 200.

Figure 71 shows the efficiency of the ontological reasoning on the SWARMs ontology with regard to a different number of instances. As can be seen in Figure 72, in general, the reasoning time increases proportionally. It is worth noting that the ontological reasoning provided by the context reasoner, essentially provided by the Jena OWL reasoner, is very efficient. Even in the case of reasoning over the SWARMs deterministic ontology with 200 instances, it only takes 250 milliseconds.

Figure 72. Comparison of ontological reasoning time with different individuals.

6.4.2. Test on rule-based reasoning

After invoking the `applyRuleBasedReasoning` function provided by the context reasoner component, the rule-based reasoning is applied on the SWARMS deterministic ontology. As stressed before, integrating rules and ontologies is a homogeneous approach. In other words, they are formalized in the same language. A noteworthy fact is that the reasoning capability provided by the Pellet reasoner actually also includes the essential ontological reasoning. In this sense, the Pellet reasoner provides both ontological and rule-based reasoning. Nonetheless, the rule-based capability provided by the Pellet reasoner is the focus of this thesis.

The reasoning result, shown in Figure 73, indicates that the rule-based reasoning is successfully applied to the SWARMS deterministic ontology. By executing the `updateOntologyWithRuleInferredData` method, newly inferred facts are stored in the SWARMS deterministic ontology.

```
<terminated> ruleBasedReasoningValidation [Java Application] C:\Program Files\Java\jdk1.8.0_131\bin\javaw.exe (Jun 13, 2017, 10:55:37 AM)
The rule-based reasoning on the SWARMS deterministic ontology took 57845 milliseconds.
The information inferred by the Pellet reasoner has been stored in the SWARMS ontology.
```

Figure 73. Rule-based reasoning test.

The SWRL rule embedded in the SWARMs ontology is considered in the rule-based reasoning. Compared with ontological reasoning, more high-level information can be inferred. For instance, a SPARQL query, listed in the following, can be executed to the SWARMs ontology in order to get the instances of class RobotCandidateForMission.

```
String query="PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>\n"+
    "PREFIX owl: <http://www.w3.org/2002/07/owl#>\n"+
    "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>\n"+
    "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>\n"+
    "PREFIX ns: <http://www.semanticweb.org/SWARMs/ontology#>\n"+
    "SELECT ?x\n"+
    "WHERE {\n"+
    "?x rdf:type ns:RobotCandidateForMission. \n"+
    "}";
```

```
Query query=QueryFactory.create(query);
QueryExecution qe=QueryExecutionFactory.create(query, mo);
ResultSet rs=qe.execSelect();
ResultSetFormatter.outputAsXML(rs);
```

The same SPARQL query is applied to the SWARMs deterministic ontology updated after the ontological and rule-based reasoning, respectively. Figure 74 shows the query result after applying the ontological reasoning. It indicates that the class RobotCandidateForMission has no instance in the updated SWARMs ontology after the ontological reasoning. An instance of class RobotCandidateForMission, robot_Alister (shown in Figure 75), is obtained after querying the SWARMs ontology updated with information based on the rule-based reasoning. The difference between the aforementioned two query results can demonstrate that the rule-based reasoning can compensate the incapability of the ontological reasoning in the specification and reasoning of rules.

```

<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="x"/>
  </head>
  <results>
  </results>
</sparql>

```

Figure 74. Query result from the SWARMs ontology after ontological reasoning.

```

<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="x"/>
  </head>
  <results>
    <result>
      <binding name="x">
        <uri>http://www.semanticweb.org/SWARMs/ontology/robot_Alister</uri>
      </binding>
    </result>
  </results>
</sparql>

```

Figure 75. Query result from the SWARMs ontology after rule-based reasoning.

As shown in Figure 73, the execution time required by the rule-based reasoning, 57845 milliseconds, is much longer than the ontological reasoning on the same knowledge base.

Apart from studying the relationship between the response time used by the rule-based reasoning and individual numbers, the effect of the number of rules to the response time is also of interest. Thus, two different sets of rules are added to the SWARMs knowledge base. In the first case, 9 more rules are added and therefore the rule set in the knowledge base consists of 10 rules. The second case increases the rule set of the knowledge base to 30 rules. As can be seen in Figure 76, in both cases, the execution time increases proportionally to the number of instances contained in the SWARMs ontology. In addition, the inference process takes more time as the number of rules increases. However, it seems that the number of instances affects the execution time to a greater extent than the number of rules.

Figure 76. The performance of rule-based reasoning.

6.4.3. Test on MEBN reasoning

The MEBN model proposed in section 4.2.6 is used to validate the usefulness and performance of the MEBN reasoning provided by the context reasoner. This MEBN model has been used to show the capabilities of the MEBN reasoning in section 5.2.3 by using the UnBBayes GUI. The aim of this section is to verify if the MEBN reasoning, programmatically provided by the context reasoner, on the same model, can get the same reasoning result.

Taking the .ubf and the .plm file generated in section 4.2.6 as an example, calling the `applyMEBNReasoning (SeverityLevel, region_1)` method can instantiate the MEBN model based on the findings in the knowledge base and the specified node (SeverityLevel) and its argument (region_1). For instance, in order to get probabilities of the severity level of region_1, the `getMEBNReasoningResult( )` method can be called.


```

<terminated> MEBNReasoningValidation [Java Application] C:\Program Files\Java\jdk1.8.0_131\bin\javaw.exe (Jun 12, 2017, 6:40:38 PM)
MEBNQUERY: LOADING MEBN FILE: src/main/resources/MEBN/severitylevelMEBNmodel.ubf
INFO: Loading triples
INFO: [ProtegeOWLParser] Completed triple loading after 268 ms
INFO: [TripleChangePostProcessor] Completed lists after 0 ms
INFO: [TripleChangePostProcessor] Completed anonymous classes after 0 ms
INFO: [TripleChangePostProcessor] Completed deprecated classes after 0 ms
INFO: [TripleChangePostProcessor] Completed properties after 22 ms
INFO: [TripleChangePostProcessor] Completed named classes after 0 ms
INFO: ... Loading completed after 306 ms
MEBNQUERY: INITIALIZING KNOWLEDGE BASE
MEBNQUERY: EXECUTING LASKEY'S ALGORITHM
Evaluate context nodes with the possibility of use reference uncertain nodes
TrySimpleEvaluationOfPossibleReferenceUncertainNodes: [( rgn = Location(oil) )]
ContextNode: ( rgn = Location(oil) )
trySearchStrategyForPossibleReferenceUncertainNodes: [( rgn = Location(oil) )]
ContextNode: ( rgn = Location(oil) )
tryUncertainReferenceStrategy [( rgn = Location(oil) )]
ContextNode: ( rgn = Location(oil) )
OVFault[oil]
SeverityLevel__region_1
Executing the SSBN takes 438 milliseconds
*****The probability distribution of the severity level of region_1*****
*****{High=0.88, Low=0.12}*****

```

Figure 77. The query result of the severity level of region_1.

Figure 77 shows the query result of the severity level of region_1. As can be seen in Figure 77, the severity level of region_1 is high with a probability of 88% and low with a probability of 12%. This reasoning result is in line with the result obtained from using UnBBayes GUI presented in section 5.2.3. In addition, it can be known that the context reasoner takes 438 milliseconds in order to make reasoning on the MEBN model. The MEBN reasoning time varies depending on the size of MEBN models. The size of an MEBN model refers to the sum of context nodes, resident nodes, and ordinary nodes. In other words, it means the number of random variables involved in a generated SSBN. Figure 78 shows the changes of executing time in terms of applying on MEBN models in different size. The MEBN reasoning time used to reason on MEBN models with 23, 28, 32, 36, and 40 nodes is recorded. As depicted in Figure 78, it takes 1482 milliseconds for the context reasoner to apply the MEBN reasoning on an MEBN model with 40 nodes.

Figure 78. The performance of the MEBN reasoning.

Compared with the rule-based reasoning, it seems that the MEBN reasoning is more affordable in terms of efficiency. The MEBN reasoning time gradually increases according to the growth of MEBN models. More detailed performance analysis on the MEBN reasoning can be referred to [164].

The three different context reasoning capabilities provided by the context reasoner are validated through some examples. The usefulness of different context reasoning can be verified. With the combination of three different context reasoning, the context reasoner is able to accommodate different reasoning requirements by providing the most suitable reasoning capability. In addition, a preliminary analysis on the performance of different context reasoning is provided. In general, the time required to execute different context reasoning methods is acceptable. The different context reasoning functions provided by the context reasoner can be selectively used by the data access manager according to requirements stemmed from different scenarios. For instance, when the SWRL rules are considered to be excluded from the reasoning process, then the ontological reasoning function provided by the context reasoner can be applied. In this way, the ontology model will only be reasoned based on the ontological reasoning algorithm.

On the contrary, when rules are important for deducing new knowledge, then the rule-based reasoning function provided by the context reasoned can be used.

7. Conclusions and future work

This chapter is dedicated to summarizing the main achievements provided in this dissertation and meanwhile pointing out the direction of study on realizing context awareness in underwater robots in the future.

7.1. Conclusions

Context awareness is an important enabler for underwater robots to know the operational environment, understand information that is exchanged between them, and adapt their behaviors accordingly to any change occurred in the dynamic and uncertain underwater environment. The context awareness feature is particularly significant when attempting to cooperate a group of underwater robots to complete complex operations or missions. The reason is that underwater robots must be able to accommodate unexpected changes due to the complexity, dynamicity, and uncertainty of the operations or missions that they are targeting to accomplish.

Aiming at facilitating context awareness in the cooperation of underwater robots, this thesis has made a set of contributions focusing on three research lines, context-aware framework, context modeling, and context reasoning.

Chapter 2 has presented a comprehensive study on the state of art related to these three topics. Specifically, it reviews the existing context modeling technologies and analyzes their advantages and disadvantages. Due to the uncertain nature of the underwater environment, it is pointed out that all of the current context modeling technologies lack the capability of representing context uncertainty in a principled way. Therefore, current solutions for modeling context uncertainty based on different theories, such as fuzzy logic and MEBN, are also investigated. Existing ontologies in the underwater robotic field have been reviewed and found to be unsuitable for representing information in the cooperation of underwater robots. Based on the aforementioned overview, ontologies have been chosen as the modeling technology for representing information in the cooperation of underwater robots. In addition, fuzzy ontology and probabilistic ontology have been highlighted to be adopted to deal with context uncertainty. Regarding the context reasoning topic, existing context reasoning methods have been reviewed and a conclusion has been drawn from the review which is a hybrid context reasoning mechanism is preferable in the cooperation of underwater robots. A set of technical features that are necessary for designing a context-aware framework has been extracted based on the analysis of existing context-aware frameworks. Specifically, the following features, semantic capability, multimodal context reasoning, context uncertainty management, ontology development methodology, architectural style, context storage *etc.*, have been considered as key features for

designing a desired context-aware framework. Lessons learned from the background and related work presented in chapter 2 have been taken as valuable inputs for this thesis to carry out new contributions to the aforementioned three research topics.

An architectural proposal of a context-aware framework for the cooperation of underwater robots has been presented in this thesis. The proposed context-aware framework aims to provide a general solution for effective context management. It consists of a set of components, including data processor, ontology model, semantic mapper, context reasoner, rules creator, and semantic query. These components are designed to provide different capabilities to achieve context awareness. The proposed framework is envisaged to deliver context-awareness and be exploited in different ways, such as enabling robots with a common understanding of the environment, allowing operators to better conceive mission plans, and optimizing communications.

Centering on the context modeling topic, three main contributions have been made in this thesis. Firstly, a new fuzzy ontology development methodology (FODM) has been presented in this thesis. The proposed methodology is dedicated to presenting the first methodological approach to building fuzzy ontologies from scratch, rather than converting existing crisp ontologies into fuzzy ones. In addition, due to its generality, it can also provide a good guide for developing crisp ontologies. Secondly, an ontology proposal, named the SWARMs ontology, has been presented to model information that is necessarily exchanged in the cooperation of underwater robots and it has been implemented for being deployed in the ontology model component. The proposed SWARMs ontology is a network of ontologies. It consists of four domain-specific ontologies, the robotic vehicle ontology, the mission & planning ontology, the environment recognition & sensing ontology, and the communications & networking ontology, which are interconnected through a core ontology. In addition, the SWARMs ontology can be extended with application specifications. When dealing with applications/scenarios where context uncertainty is involved in, it can incorporate fuzzy extensions or probabilistic extensions. Specifically, the adoption of fuzzy ontology aims to deal with context vagueness and ambiguity while the probabilistic ontology is able to represent context inaccuracy, incompleteness, and randomness based on the MEBN theory. The SWARMs ontology, referring to the general ontology model which includes the deterministic part and may include fuzzy or probabilistic extensions depending on the target scenario, can provide a comprehensive and principled representation for the information exchanged between robots and also provide substantial support for uncertainty reasoning, such as the MEBN reasoning. Lastly, the first application of Stochastic Reduced Order Model (SROM) to quantify uncertainty propagated in mathematics-embedded

ontologies has been presented. A use case regarding the SWARMs ontology has been used to illustrate the usefulness of the proposal. The proposal has been proven to be able to accurately approximate the statistics of uncertain entities with less computation. This approach is very general and can be applied to ontologies with any kind of mathematic relationship.

Focusing on the context reasoning field, this thesis has proposed a hybrid context reasoning mechanism which integrates the ontological, rule-based, and MEBN reasoning methods. With the combination of different reasoning methods, this proposal can enhance the overall performance by mitigating each one's weakness using others' strengths. The proposed reasoning mechanism has been validated in two ways. On one hand, a preliminary proof of concept has been done to testify the capabilities of three different reasoning methods by using existing graphical tools, such as the Protégé and the UnBBayes GUI. On the other hand, a set of Java APIs has been developed to implement the proposed reasoning mechanism in the context reasoner component by using existing context reasoners, the Jena OWL, Pellet, and UnBBayes-MEBN built-in reasoner. The context reasoner has been validated by using some examples in terms of its usefulness. A quantitative analysis on its performance has also been provided. The validation result has shown that different context reasoning capabilities provided by the context reasoner can be useful to reason about context information and its uncertainty in the cooperation of underwater robots with an accepted execution time.

7.2. Future work

Based upon the conclusions above and considering the limitations of the work existed, future research can be carried out in the following areas:

- The SWARMs ontology is acknowledged to adopt fuzzy ontologies to represent vague and ambiguous context information. Considering it is not the focus of this thesis, only a simple use case on seabed characterization is presented to show how the SWARMs ontology is able to be extended with fuzzy specifications. The fuzzy ontology is not included in the further implementation and validation of the SWARMs ontology. Future work can be put on extending the use of fuzzy ontologies in the SWARMs ontology to deal with scenarios or applications that involve context vagueness and ambiguity.
- The usefulness of the SROM-based uncertainty quantification approach has been demonstrated through a use case. Future efforts can be made to implement this algorithm and produce a general toolkit to be used in any mathematics-embedded ontologies for quantifying propagated uncertainty.

- Currently, the context reasoner only integrates the ontological, rule-based, and MEBN reasoning. Introducing fuzzy logic reasoning into the context reasoner could be a further improvement. As introduced in chapter 2.1.2, the fuzzy DL reasoner, developed by Bobillo *et al.* [66], can provide fuzzy logic reasoning based on fuzzy ontologies. Future work can focus on integrating the fuzzy DL reasoner in the context reasoner in order to provide corresponding reasoning capabilities for fuzzy ontologies.
- The current design for ontology model in this thesis separates the storage of the SWARMS ontology into two parts: the deterministic part and the probabilistic part. In this way, data access manager can directly update the knowledge base of the probabilistic part and allow the MEBN reasoning provided by the context reasoner to directly reason on the MEBN model, namely the probabilistic ontology. Pablo [164] proposed another solution for updating the knowledge base of the probabilistic ontology. Specifically, he assumed that all context information is deterministic and he built a general deterministic ontology to represent that information. When dealing with scenarios that involve context uncertainty, he pre-built MEBN models and updated the probabilistic knowledge base by querying information from the deterministic ontology. A set of methods is developed to obtain information, which is necessary for instantiating the MEBN models, from the deterministic knowledge base. This approach can be an alternative for updating the probabilistic knowledge base. Future work can focus on investigating the possibility of adopting the approach proposed by Pablo in the context-aware framework so that the framework could be more flexible to provide different methods for updating the knowledge base.
- Due to that currently other framework components are still under development, the validation of context reasoner is directly applied to the SWARMS ontology in-memory, including the deterministic and probabilistic part. Integrating the context reasoner and ontology model with other framework components and middleware components must be realized in the near future. The integration work between context reasoner and ontology model and other middleware components might imply new requirements/changes to the context reasoner and ontology model. A further refinement of the design of context reasoner and ontology model should be taken into considerations.
- As stressed in chapter 6.4, the implementation of the context reasoner focuses on providing simple interfaces for accessing context reasoning capabilities provided by the existing context reasoners, including the Jena OWL, Pellet, and UnBBayes-MEBN built-in reasoner. In this sense, the performance of the context reasoner is essentially subject to the performance of those existing context reasoners. In the future, the foundational

theories behind those context reasoners should be thoroughly studied and efforts should be made to refine their algorithms and further improve their performance.

- Once the context-aware framework is fully implemented, it must be deployed to real usages and be tested with more real-world applications.
- It is true that the context-aware framework is conceived to be used in the underwater robot field. However, the possibility of applying it to other research fields, such as Smart Agriculture and Smart Grid, should also be explored.

7.3. Publications and projects

7.3.1. List of publication

The results presented in this thesis were published in the following journals indexed in Journal Citation Report (JCR) and national/international conferences.

1. **Xin Li**, José-Fernán Martínez, Martina Eckert, and Gregorio Rubio. Uncertainty Quantification in Mathematics-Embedded Ontologies Using Stochastic Reduced Order Model. *IEEE Transactions on Knowledge and Data Engineering*. 29, 4 (April 2017), 912–920. DOI:<https://doi.org/10.1109/TKDE.2017.2651819>. (JCR Q1, Impact Factor = 3.438)
2. **Xin Li**, Sonia Bilbao, Tamara Martínez-Wanton, Joaquim Bastos, and Jonathan Rodriguez. SWARMS Ontology: A Common Information Model for the Cooperation of Underwater Robots. *Sensors* 17, 3 (March 2017), 569. DOI:<https://doi.org/10.3390/s17030569>. (JCR Q1, Impact Factor = 2.677)
3. **Xin Li**, José-Fernán Martínez, and Gregorio Rubio. Towards a Hybrid Approach to Context Reasoning for Underwater Robots. *Applied Sciences-Basel*. 7, 2 (February 2017), 183. DOI:<https://doi.org/10.3390/app7020183>. (JCR Q3, Impact Factor = 1.679)
4. **Xin Li**, José-Fernán Martínez, and Gregorio Rubio. A New Fuzzy Ontology Development Methodology (FODM) Proposal. *IEEE Access* 4 (2016), 7111–7124. DOI:<https://doi.org/10.1109/ACCESS.2016.2621756>. (JCR Q1, Impact Factor = 3.244)
5. Gregorio Rubio, José-Fernán Martínez, David Gómez, and **Xin Li**. Semantic Registration and Discovery System of Subsystems and Services within an Interoperable Coordination Platform in Smart Cities. *Sensors* 16, 7 (June 2016), 955. DOI:<https://doi.org/10.3390/s16070955>. (JCR Q1, Impact Factor = 2.677)
6. **Xin Li**, José-Fernán Martínez, Jesús Rodríguez-Molina, and Néstor Lucas Martínez. A Survey on Intermediation Architectures for Underwater Robotics. *Sensors* 16, 2 (February 2016), 190. DOI:<https://doi.org/10.3390/s16020190>. (JCR Q1, Impact Factor = 2.677)
7. **Xin Li**, Martina Eckert, José-Fernán Martínez, and Gregorio Rubio. Context Aware Middleware Architectures: Survey and Challenges. *Sensors* 15, 8 (August 2015), 20570–20607. DOI:<https://doi.org/10.3390/s150820570>. (JCR Q1, Impact Factor = 2.677)
8. **Xin Li**, José-Fernán Martínez, and Gregorio Rubio. Context Reasoning in Underwater Robots Using Multi-Entity Bayesian Network. *ICCR 2016: International Conference on Cloud and Robotics*, 22-24 November 2016, Saint-Quentin, France. (**Best Student Paper Award**)

9. Tamara Martínez-Wanton, Elena Muelas, Raúl Santos de la Cámara, Jesús Rodríguez-Molina, **Xin Li**, and José-Fernán Martínez. Modelado de vehículos autónomos y la incertidumbre de su entorno para la seguridad de operaciones marítimas. *IV Congreso Nacional de I+D en Defensa y Seguridad*, 16-18 November 2016, San Javier, Murcia, Spain.

7.3.2. Research projects

- **European projects**
 - **ACCUS** - Adaptive Cooperative Control in Urban (sub)Systems
 - **SWARMS** – Smart and Networking Underwater Robots in Cooperation Meshes

Appendix A: Ontology descriptions

Table 9, Table 10, and Table 11 shows the main elements, including classes, object properties, and data properties, defined in the Robotic Vehicles domain-specific ontology, respectively.

Table 9. Concepts defined in the robotic vehicle ontology

Concept name	Synonyms	Description
Vehicle		A vehicle models any platforms in or by which someone travels or something is carried or conveyed
MotorizedVehicle		Motorized vehicles have propulsion system on-board. All ROVs, USVs, and vessels are motorized and some AUVs and ASV are motorized.
ROV	Remotely Operated Vehicle	A Remotely Operated Vehicle (ROV) is a robotic vehicle operated by a human operator from a location off the robot (mainly in a vessel) through a tether. It operates under the water. The robot takes no initiative and relies on continuous or nearly continuous input from the human operator. The energy for the propulsion system (an ROV is motorized) can be either in or off the vehicle (cable is then also used to transfer the power to the vehicle).
USV	Unmanned Surface Vehicle	An Unmanned Surface Vehicle (USV) is a robotic vehicle with a propulsion system (then motorized) that maneuvers on the surface of the water (without a tether) and is remotely controlled by a pilot or programmed to navigate a predefined course or mission (then automated robot).
Vessel		A vessel is a vehicle with propulsion (then motorized) operating on and restricted to the surface of a water body. It is specifically equipped, manned and operated for scientific, usually oceanographic, research.
RoboticVehicle		A robotic vehicle is a physical object designed by a human agent to provide a service by acting here on the water domain.
SurfaceRobot		Surface robots are surface and robotic vehicles.
ASV	Autonomous Surface Vehicle	An Autonomous Surface Vehicle (ASV) is a robotic vehicle with a propulsion system (then motorized) or not (then unmotorized as the surface glider) that has the capability to perform high-level tasks with the limited intervention of an operator. It operates on the surface of the water.
SurfaceGlider		
UnderwaterRobot		Underwater robots are underwater and robotic vehicles.
AUV	Autonomous Underwater Vehicle	An Autonomous Underwater Vehicle (AUV) is a robotic vehicle with a propulsion system (then motorized) or not (then unmotorized as the underwater glider) that has the capability to perform high-level tasks with the limited intervention of an operator. It operates under the water.
SurfaceVehicle		Surface vehicles move only at the surface of the water (unlike underwater vehicles).
SurfaceRobot		Surface robots are surface and robotic vehicles.

UnderwaterVehicle		Underwater vehicles can move under the water, fish-like or submarine-like vehicles.
UnmotorizedVehicle		Unmotorized vehicles have no propulsion system on-board. Either they use the environment (like gliders) or they stay in the water after their field installation (like buoys).
Buoy		A buoy is an unmotorized float collecting weather and ocean data. There exist moored and drifting buoys.
VehicleCapabilityType		This concept represent the different types of capabilities
VehicleCapability		This information describes both the vehicle's features and its abilities, taking into account that the latter can change, e.g. in the case of insufficient power level or specific module failure. Therefore, it indicates the vehicle specifications, like available (operational) sensors, and the operating system it uses, gathering all the required metadata in order to recognize what can be achieved with such particular AUV or ROV, and how.

Table 10. Relations defined in the robotic vehicle ontology

Object property	Domain	Range	Inverse relation
canPerform	Vehicle	Action	performedBy
canReplace	RoboticVehicle	RoboticVehicle	
fasterThan	Robot	Robot	
hasLessAutonomythan	RoboticVehicle	RoboticVehicle	
hasMoreAutonomyThan	RoboticVehicle	RoboticVehicle	
hasAcousticPosition	SensingData or Sensor or Vehicle	AcousticPosition	
hasGPSPosition	Region or SensingData or Sensor or Vehicle	GPSPosition	
hasInertiaPosition	SensingData or Sensor or Vehicle	InertiaPosition	
hasSensor	Infrastructure or RoboticVehicle or Vessel	Sensor	
hasType	VehicleCapability	VehicleCapabilityType	
hasVehicleCapability	RoboticVehicle or Vehicle	VehicleCapability	
hasPriority	Task, CommunicationLink	Priority	isPriorityOf
hasTaskSpecification	Task	TaskSpecification	isTaskStatusOf
hasTaskStatus	Action or Task	TaskStatus	isTaskStatusOf

Table 11. Data properties defined in the robotic vehicle ontology

Data property	Domain	Range
consumption	Vehicle or VehicleCapability	Double
description	Task or Vehicle or VehicleCapability or VehicleCapabilityType	String

distanceToSeabed	RoboticVehicle	Int
id	Action or Mission or Task or Vehicle or VehicleCapability	Int
robotSpeed	Robot	Short
vehicleBatteryLevel	RoboticVehicle	Float
vehicleCurrentSpeed	Vehicle	Double
vehicleLength	Vehicle	Int
vehicleMaxBattery	Vehicle	Int
vehicleName	Vehicle	String
vehicleNominalSpeed	Vehicle	Double
vehicleType	Vehicle	String
workingDepth	RoboticVehicle	UnsignedShort

Table 12, Table 13, and Table 14 show the main elements, including classes, relations, and data properties, defined in the mission & planning domain-specific ontology, respectively.

Table 12. Concepts defined in the mission & planning ontology

Concept	Synonyms	Description
Action		An action is an elementary task toward vehicle controllers/sensors. For instance, the vehicle level task “SurveyAnArea” is vehicle planned as a sequence of “trackPattern” actions.
Event		An event is something that occurs in a certain place during a particular time with a significant relevance to be notified to the involved agents.
AlarmEvent		An occurrence of type alarm.
ChangeEvent		An occurrence of a modification in a task, in the environment, <i>etc.</i>
RequestEvent		An occurrence of a request from an agent in the system.
Person	foaf:Person	It is the general definition of people who may be involved in the mission planning process.
Operator		A subclass of Person, it characterizes a group of people who are responsible for operating vehicles.
Mission		A mission is the input to the mission planner component. The mission will be provided by an operator and is defined as a set of goals to be performed by a swarm of vehicles (AUV, ROV, USV).
MissionPlan		A mission plan is a sequence of scheduled low-level tasks (operator and vehicle level tasks) that need to be carried out to achieve a mission, with dependencies between tasks and approximate time duration.
MissionOptimizationCriteria		Criteria to be used by the algorithms to optimize the mission, <i>e.g.</i> time, battery, cost.
Priority		It allows defining different priorities for the tasks or actions to be executed by the vehicles.
Service		Any functionality or capability provided by assets, such as vehicles, or communication facilities, can be regarded as services.

Task		Tasks are ways to change the system's state and achieve specific objectives.
TaskStatus		It defines the status of tasks, possible instances could be Healthy, Stopped, etc.
TaskRegionType		The region where a task is executed. It could be a point, a row or an area described by a polygon.
VehiclePlan		A vehicle plan is a sequence of operations to be carried out by a vehicle in order to accomplish a vehicle level task. It mimics the mission plan at the vehicle level.

Table 13. Object properties defined in the mission & planning ontology

Object property	Domain	Range	Inverse relation
actionArea	Action	Region	
addressee	Message	Target	
availableswVehicles	Mission	RoboticVehicle	
Bearing	Action	InertiaPosition	
belongsToVehiclePlan	Action	VehiclePlan	containsAction
containsTask	MissionPlan	LowLevelTask	belongsToMissionPlan
forbiddenArea	Mission	Region	
isAchievedBy	Mission	Action	aimsFor
achievesMision	MissionPlan	Mission	isAchievedByMissionPlan
hasConstraint	Mission	MissionConstraint	isConstraintOf
hasDescription	Mission	MissionDescription	isDescriptionOf
hasDuration	Mission	MissionDuration	isDurationOf
hasGoal	Mission		isGoalOf
hasMissionSpecification	Mission	MissionSpecification	
hasOptimizationCriteria	Mission	MissionSpecificationCriteria	isOptimizationCriteriaOf
navigationArea	Mission	Region	
Overlap	Seabed	Seabed	
achievesTask	VehiclePlan	VehicleLevelTask	isAchievedByVehiclePlan
allocatedTo	OperatorLevelTask	Operator	
assignedTo	VehicleLevelTask	Vehicle	performs
belongsToMissionPlan	LowLevelTask	MissionPlan	containsTask
forms	Task	HighLevelTask	isComposedOf
hasPriority	Task CommunicationLink	Priority	isPriorityOf
hasTaskSpecification	Task	TaskSpecification	isTaskStatusOf
hasTaskStatus	Task or Action	TaskStatus	isTaskStatusOf
requires	Task	VehicleCapabilityType	
taskType	Task	TaskRegionType	

Table 14. Data properties defined in the mission & planning ontology

Data property	Domain	Range
endTime	Action	Int
gpsAltitude	Action, GPSPosition	Float
id	Action or Mission or Task or Vehicle or VehicleCapability	Int
maxSpeed	Task	Double
missionDescription	MissionDescription	String
missionDuration	MissionDuration	Double
range	Action	Float
startTime	Action	Int
timeLapse	Action	Int

Table 15, Table 16, and Table 17 show the main elements, including classes, object properties, and data properties, defined in the environment recognition & sensing domain-specific ontology, respectively.

Table 15. Concepts defined in the environment recognition & sensing ontology

Concept name	Description
DataProcessor	Any processing operation that is necessary to perform on sensed data concerning the environment or any Entity involved in a SWARMS mission is modeled in the scope of this concept.
Entity	This concept represents any being or object, fixed or dynamic, to be found at the seabed, water column or surface.
BioticEntity	This concept represents the living entities of the marine ecosystem.
ManmadeEntity	It represents any entity that is not made by nature and is thus manmade.
Infrastructure	Manmade infrastructure for the exploration of offshore resources, with its base planted at the seabed.
OtherObject	Any manmade object that doesn't integrate any sensor, and thus cannot provide any measurement data.
Landmark	Specific seabed feature, object or structure fixed on the seabed, which presence can be easily detected or extracted from existing or collected imaging data, through image processing techniques.
ProcessedData	The concept encompasses any sensed data that was submitted to processing, on the environment or concerning any Entity.
Region	A bi-dimensional area projected on the seabed, within a defined perimeter.

Seabed	Underwater surface at the bottom of the maritime water column
SensingData	The concept encompasses the data that is sensed and provided by physical and logical sensors, on the environment or concerning the Entity they are mounted on.
Image	Data type composed of an array of colored pixels.
Pollutant	This class represents materials and/or any chemical substance in the considered environment, which shouldn't be present and/or can be harmful to life in the considered ecosystem.
Position	Position is the super class of GPS and InertiaPosition and represents a georeferenced location that can be associated to an Entity to characterize its location.
AcousticPosition	Position estimated through acoustic means.
GPSPosition	Position estimated through GPS means (satellites, receiver, chipset, signals, <i>etc.</i>).
InertiaPosition	Inertial navigation is a self-contained navigation technique in which measurements provided by accelerometers and gyroscopes are used to track the position and orientation of an object relative to a known starting point, orientation and velocity. Is estimated through inertial sensors (e.g. IMU chip in robotic vehicles).
Pressure	This represents the measured pressure in the water column, or at the surface (atmospheric pressure).
Sound	This class represents the speed of sound, which expresses the distance traveled per unit time by an acoustic wave propagating through a certain medium, i.e. salt water in SWARMs case.
TemporalReference	It represents the time at which a specific event occurred, e.g. a measurement or an action.
WaterConductivity	It represents water conductivity at a specific position, depth and time.
WaterCurrent	Currents define the displacement of masses of water in the ocean and are usually represented by the direction toward which they flow (Set) and their speed (Drift).
WaterSalinity	It represents water salinity at a specific position, depth and time.
WaterTemperature	It represents water temperature at a specific position, depth and time. Temperature is one of the most measured characteristics in the ocean. In general, for most areas of the ocean, the water temperature decreases from the surface to the bottom.
WaterTurbidity	It is the thickness or opaqueness of water caused by the suspension of the matter. Turbidity is a dimensionless quantity which is expressed in NTU (Nephelometric Turbidity Units).
Wave	It represents a wave with a particular heading, speed, and height, at the surface of the water column.
Wind	It represents the wind with a particular heading, speed, and variability, at the surface of the water column.

Sensor	A sensor typically provides physical measurements of the environment where it is located, but can also be considered as the source of other logical data representative of an environment, as well as of the current conditions and/or status of a (robotic) vehicle. A sensor can be either physical or logical. Therefore, two subclasses of Sensor class have been defined.
LogicalSensor	It represents logical or virtual sources of context data, e.g. Weather Web Service.
PhysicalSensor	Represents a sensor from which physical measurements can be obtained, e.g. temperature, location, which are suitable for processing afterward.
WaterColumn	This concept represents the mass of water surrounding the robotic vehicles involved in a mission, delimited by the seabed and the water surface.
WaterSurface	This concept represents the surface above the water column, where vessels, buoys, and some robotic vehicles can operate while involved in a mission.

Table 16. Object properties defined in the environment recognition & sensing ontology

Object property	Domain	Range	Inverse relation
adjacentTo	Seabed	Seabed	
attachedTo	Sensor	Infrastructure or RoboticVehicle or Vessel	
below	Entity	Entity	
biggerThan	Entity	Entity	
canBeProcessedBy	SensingData	DataProcessor	
canProcess	DataProcessor	SensingData	isProcessedBy
entityCloseTo	Entity	Entity	
seabedCloseTo	Seabed	Seabed	
colocated	ProcessedData	ProcessedData	
compilesData	ProcessedData	SensingData	
entityFarFrom	Entity	Entity	
seabedFarFrom	Seabed	Seabed	
hasConductivity	WaterColumn	WaterConductivity	
hasEntity	Seabed or WaterColumn or WaterSurface	Entity	
hasHigherAccuracyThan	Sensor	Sensor	
hashigherResolutionThan	Sensor	Sensor	
hasLandmark	Seabed or WaterColumn or WaterSurface	Landmark	
hasLowerAccuracyThan	Sensor	Sensor	
hasLowerResolution	Sensor	Sensor	

hasAcousticPosition	SensingData or Sensor or Vehicle	AcousticPosition	
hasGPSPosition	Region or SensingData or Sensor or Vehicle	GPSPosition	
hasInertiaPosition	SensingData or Sensor or Vehicle	InertiaPosition	
hasPressure	Seabed or WaterColumn	Pressure	
hasSensor	Infrastructure or RoboticVehicle or Vessel	Sensor	
hasSensorHost	Sensor	Infrastructure or RoboticVehicle or Vessel	
hasSound	WaterColumn	Sound	
hasTemporalReference	ProcessedData or SensingData	TemporalReference	
hasWaterCurrent	WaterColumn or WaterSurface	WaterCurrent	
HasWaterSalinity	WaterColumn	WaterSalinity	
hasWaterTemperature	WaterColumn or WaterSurface	WaterTemperature	
hasWaterTurbidity	WaterColumn or WaterSurface	WaterTurbidity	
hasWave	WaterSurface	Wave	
hasWind	WaterSurface	Wind	
heavierThan	Entity	Entity	
higherThan	ProcessedData	ProcessedData	
inside	Entity	Entity	
inOn	Landmark	Seabed	
isSimultaneousTo	ProcessedData	ProcessedData	
lighterThan	Entity	Entity	
Overlap	Seabed	Seabed	
performsClassificationOf	DataProcessor	SensingData	
performsFeatureExtractionOn	DataProcessor	ProcessedData	
performsProcessingOf	DataProcessor	SensingData	
performsRecognitionOn	DataProcessor	ProcessedData	
provides	DataProcessor	ProcessedData	
refersAfter	ProcessedData	ProcessedData	
refersAfter	SensingData	SensingData	
sensedBefore	SensingData	SensingData	
sensedSimultaneously	SensingData	SensingData	
similarTo	Entity	Entity	
slowerThan	Entity	Entity	

smallerThan	Entity	Entity	
Source	DataProcessor	ProcessedData	
surfacePoints	Position	WaterSurface	
allocatedTo	Sensor	Infrastructure or RoboticVehicle or Vessel	
usesModem	Infrastructure or RoboticVehicle or Vessel	Infrastructure or RoboticVehicle or Vessel	

Table 17. Data properties defined in the environment recognition & sensing ontology

Data property	Domain	Range
depth	Seabed or WaterColumn	Float
dopplerVelocity		unsignedByte
entityComposition	Entity	Short
entitySize	Entity	Short
entityType	Entity	Byte
gpsAltitude	Action, GPSPosition	Float
gpsDepth	GPSPosition or Landmark	Double
gpsHeadig	GPSPosition	Float
gpsLatitude	GPSPosition or Landmark	Float
gpsLongitude	GPSPosition or Landmark	Float
h2sPollution	Pollutant	unisgnedShort
imageCoding	Image	String
imageData	Image	Byte
imageDepth	Image	Int
imageHeight	Image	Int
Imagesize	Image	Int
imageWidth	Image	Int
infrastructureType	Infrastructure	Byte
landmarkType	Landmark	Literal
pitch	InertiaPosition	Float
roll	InertiaPosition	Float
yaw	InertiaPosition	Float
pressure	Pressure	float
seabedHumidity	Seabed	Short
seabedPlasticity	Seabed	Short
seabedPorosity	Seabed	Short
seabedSandType	Seabed	Byte
seabedSedimentLayerThickness	Seabed	Short
SeabedType	Seabed	String

sensorAccuracy	Sensor	byte
sensorHost	Sensor	String
sensorId	Sensor	Short
sensorResolution	Sensor	byte
sensorState	Sensor	unsignedShort
soundVelocity	Sound	unsignedShort
temporalReference	TemporalReference	dateTimestamp
waterconductivity	WaterConductivity	byte
waterCurrentDirection	WaterCurrent	string
waterCurrentVelocity	WaterCurrent	unsignedByte
waterSalinity	WaterSalinity	unisgnedShort
waterTemperature	WaterTemperature	Byte
waterTurbidity	WaterTurbidity	unsignedShort
waterVelocityMagnitude	WaterCurrent	unsignedShort
waveHeight	Wave	Short
wavePeriod	Wave	Short
windDirection	Wind	byte
windHeight	Wind	Short
windPowerDensity	Wind	Short
windSpeed	Wind	Short
xcoordinate	InertiaPosition	Float
ycoordinate	InertiaPosition	Float
zcoordinate	InertiaPosition	Float

Table 18, Table 19, and Table 20 show the main elements, including classes, object properties, and data properties, defined in the communication & networking domain-specific ontology, respectively.

Table 18. Concepts defined in the communication & networking ontology

Concept name	Description
AcousticModem	An acoustic modem. It offers the possibility of wireless communication under water.
CommunicationLink	A communications channel that connects two or more devices. This link may be an actual physical link or it may be a logical link that uses one or more actual physical links.
Acoustic	An acoustic communication channel.
Cable	A communication channel that uses wires of cables to transmit data and information.

Radio	A radio communication system. It sends signal by radio.
Satellite	A satellite communication system. It sends signals by satellite.
Wifi	A Wifi communication system.
Message	A unit of communication intended by the source for consumption by some recipient or group of recipients.
ErrorMsg	A message that informs about an error.
NotificationMsg	A message that contains a notification
QueryMsg	A message that contains a query.
RegistrationMsg	A message that contains a request for registration in the system
RequestMsg	A message that contains a generic request.
Source	The sender of the message
Target	The recipient of the message.

Table 19. Object properties defined in the communication & networking ontology

Relation name	Domain	Range	Inverse relation
addressee	Message	Target	
source	Message	Source	
usesModem	Acoustic	AcousticModem	
hasPriority	CommunicationLink	Priority	isPriorityOf

Table 20. Data properties defined in the communication & sensing ontology

Data property	Domain	Range
hasTimestamp	Message	Long
headerSubType	Message	String
headerType	Message	String
messageContent	Message	String
RxPower	Radio	integer
TxPower	Radio	Integer
state	Radio	Boolean (1 = active, 0 not active)
lastUpdate	Radio	dateTimestamp

Appendix B: Reasoning APIs

The classes that are used to manage the reasoning functionalities can be seen in Figure 79, Figure 80, and Figure 81. Specifically, three classes, `OntologicalReasoner`, `RuleBasedReasoner`, and `MEBNReasoner`, are created to implement the `ontologicalReasoning`, `ruleBasedReasoning`, and `mebnReasoning` interfaces, respectively.

Figure 79. APIs of the `OntologicalReasoner` class.

reasoners

Class RuleBasedReasoner

java.lang.Object

reasoners.RuleBasedReasoner

All Implemented Interfaces:

ruleBasedReasoning

public class RuleBasedReasoner

extends java.lang.Object

implements ruleBasedReasoning

Constructor Summary

Constructors

Constructor and Description

RuleBasedReasoner()

Method Detail

applyRuleBasedReasoning

public com.hp.hpl.jena.rdf.model.InfModel applyRuleBasedReasoning(com.hp.hpl.jena.rdf.model.Model m)

Specified by:

applyRuleBasedReasoning in interface ruleBasedReasoning

Parameters:

m - m is the model that is going to be reasoned.

Returns:

InfModel This method applies the rule-based reasoning to the specified ontology model.

updateOntologyWithRuleInferredData

public void updateOntologyWithRuleInferredData(com.hp.hpl.jena.rdf.model.InfModel m3,
com.hp.hpl.jena.rdf.model.Model m4)

Specified by:

updateOntologyWithRuleInferredData in interface ruleBasedReasoning

Parameters:

m3 - The inferred model

m4 - the original model This method updates the ontology model with the rule-based reasoning results.

Figure 80. APIs of the RuleBasedReasoner class.

reasoners

Class MEBNReasoner

java.lang.Object
reasoners.MEBNReasoner

All Implemented Interfaces:

mebnReasoning

```
public class MEBNReasoner
extends java.lang.Object
implements mebnReasoning
```

Constructor Detail

MEBNReasoner

```
public MEBNReasoner(java.lang.String mebnFile,
                    java.lang.String findingFile)
    throws java.io.IOException
```

Constructor

Parameters:

mebnFile - String MEBN .ubf file path

findingFile - String finding .plm file path

Throws:

java.io.IOException - throws an IO exception if the mebn file and finding file are not found.

Method Detail

applyMEBNReasoning

```
public void applyMEBNReasoning(java.lang.String node,
                              java.lang.String argument)
    throws java.lang.Exception
```

Specified by:

applyMEBNReasoning in interface mebnReasoning

Parameters:

node - String specify the random node that is of interest

argument - String specify the specific instance of the entity that is the argument of the random node

Throws:

java.lang.Exception - Laskey's algorithm error

getMEBNReasoningResult

```
public unbbayes.prs.bn.ProbabilisticNode getMEBNReasoningResult()
```

Specified by:

getMEBNReasoningResult in interface mebnReasoning

Returns:

ProbabilisticNode this the random node that is of interest in terms of its states

Figure 81. APIs of the MEBNReasoner class.

References

- [1] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context Aware Computing for The Internet of Things: A Survey," *IEEE Commun. Surv. Tutor.*, vol. 16, no. 1, pp. 414–454, 2014.
- [2] R. Want, A. Hopper, V. Falcão, and J. Gibbons, "The active badge location system," *ACM Trans. Inf. Syst.*, vol. 10, no. 1, pp. 91–102, Jan. 1992.
- [3] J. Pascoe, "Adding generic contextual capabilities to wearable computers," in *Proceedings of Second International Symposium on Wearable Computers*, Pittsburgh, PA, USA, 1998, pp. 92–99.
- [4] A. K. Dey, "Context-aware computing: The cyberdesk project," in *Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments*, Menlo Park, CA, USA, 1998.
- [5] A. K. Dey, G. D. Abowd, and A. Wood, "CyberDesk: a framework for providing self-integrating context-aware services," in *Proceedings of the 3rd International Conference on Intelligent User Interfaces*, San Francisco, California, USA, 1998, pp. 47–54.
- [6] D. Salber, A. K. Dey, and G. D. Abowd, "Ubiquitous Computing: Defining an Hci Research Agenda for an Emerging Interaction Paradigm," GVU Technical Report git-gvu-98-1, Atlanta, USA, Georgia Institute of Technology.
- [7] P. J. Brown, "Triggering information by context," *Pers. Technol.*, vol. 2, no. 1, pp. 18–27, Mar. 1998.
- [8] N. Ryan, "Mobile Computing in a Fieldwork Environment: Metadata Elements," Project working document, version 0.2 (1997).
- [9] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," in *HUC'99 Proceeding of the 1st International symposium on Handheld and Ubiquitous Computing*, Karlsruhe, Germany, 1999.
- [10] H. van Kranenburg, M. S. Bargh, S. Iacob, and A. Peddemors, "A context management framework for supporting context-aware distributed applications," *IEEE Commun. Mag.*, vol. 44, no. 8, pp. 67–74, Aug. 2006.
- [11] P. Hu, J. Indulska, and R. Robinson, "An Autonomic Context Management System for Pervasive Computing," in *Proceedings of Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom 2008)*, Hong Kong, China, 2008, pp. 213–223.
- [12] X. Li, M. Eckert, J.-F. Martinez, and G. Rubio, "Context Aware Middleware Architectures: Survey and Challenges," *Sensors*, vol. 15, no. 8, pp. 20570–20607, Aug. 2015.
- [13] G. G. Acosta, O. A. C. Ibanez, H. J. Curti, and A. F. Rozenfeld, "Low-cost Autonomous Underwater Vehicle for pipeline and cable inspections," in *Proceedings of Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies*, Tokyo, Japan, 2007, pp. 331–336.
- [14] A. Diercks *et al.*, "NIUST - Deepwater horizon oil spill response cruise," in *Proceedings of OCEANS 2010*, Seattle, WA, USA, 2010, pp. 1–7.
- [15] C. Kaminski *et al.*, "12 days under ice-an historic AUV deployment in the Canadian High Arctic," in *Proceedings of 2010 IEEE/OES Autonomous Underwater Vehicles (AUV)*, Monterey, CA, USA, 2010, pp. 1–11.
- [16] F. Maurelli, Z. Saigol, and D. Lane, "Cognitive knowledge representation under uncertainty for autonomous underwater vehicles," in *Proceedings of IEEE ICRA'14 IEEE Hong Kong*, Kong Kong, China, 2014.
- [17] X. Li, J.-F. Martínez, J. Rodríguez-Molina, and N. Martínez, "A Survey on Intermediation Architectures for Underwater Robotics," *Sensors*, vol. 16, no. 2, p. 190, Feb. 2016.

- [18] The Pandora project. Available: <http://persistentautonomy.com/>. (Accessed on 21st June 2017)
- [19] The TRIDENT project. Available: <http://www.irs.uji.es/trident/>. (Accessed on 21st June 2017)
- [20] The CoCoRo project. Available: <https://ec.europa.eu/digital-single-market/en/news/cocoro-robot-swarms-use-collective-cognition-perform-tasks>. (Accessed on 21st June 2017)
- [21] The SWARMS project. Available: <http://www.swarms.eu/>. (Accessed on 21st June 2017)
- [22] Extensible Markup Language (XML). Available: <https://www.w3.org/XML/>.
- [23] JavaScript Object Notation (JSON). Available: <http://www.json.org/>.
- [24] F. Bobillo and U. Straccia, "Fuzzy ontology representation using OWL 2," *Int. J. Approx. Reason.*, vol. 52, no. 7, pp. 1073–1094, Oct. 2011.
- [25] K. B. Laskey, "MEBN: A Logic for Open-World Probabilistic Reasoning," GMU C4I Center Technical Report C4I-06-01.
- [26] S. Sarkar, J. E. Warner, W. Aquino, and M. D. Grigoriu, "Stochastic reduced order models for uncertainty quantification of intergranular corrosion rates," *Corros. Sci.*, vol. 80, pp. 257–268, Mar. 2014.
- [27] B. Schilit, N. Adams, and R. Want, "Context-Aware Computing Applications," in *Proceedings of First Workshop on Mobile Computing Systems and Applications (WMCSA)*, Santa Cruz, California, USA, 1994, pp. 85–90.
- [28] Unified Modeling Language (UML). Available: <http://www.uml.org/>.
- [29] Entity Relationship Model (ERM). Available: <http://vulms.vu.edu.pk/Courses/CS619/Downloads/CarMatch%20ERD.pdf>.
- [30] Object Role Model (ORM). Available: <http://www.orm.net/>.
- [31] K. Cheverst, K. Mitchell, and N. Davies, "Design of an object model for a context sensitive tourist GUIDE," *Comput. Graph.*, vol. 23, no. 6, pp. 883–891, Dec. 1999.
- [32] C. Xu and S. C. Cheung, "Inconsistency detection and resolution for context-aware middleware support," *ACM SIGSOFT Softw. Eng. Notes*, vol. 30, no. 5, p. 336, Sep. 2005.
- [33] N. Bradley and M. Dunlop, "Toward a Multidisciplinary Model of Context to Support Context-Aware Computing," *Hum.-Comput. Interact.*, vol. 20, no. 4, pp. 403–446, Dec. 2005.
- [34] G. Castelli, A. Rosi, M. Mamei, and F. Zambonelli, "A Simple Model and Infrastructure for Context-Aware Browsing of the World," in *Proceedings of Fifth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'07)*, White Plains, NY, USA, 2007, pp. 229–238.
- [35] D. Hong, H. R. Schmidtke, and W. Woo, "Linking context modelling and contextual reasoning," in *Proceedings of 4th International Workshop on Modeling and Reasoning in Context (MRC)*, Roskilde, Denmark, 2007, pp. 37–48.
- [36] A. Ikram, N. Baker, M. Knappmeyer, E. S. Reetz, and R. Tonjesy, "An artificial chemistry based framework for personal and social context aware smart spaces," in *Proceedings of 7th International Wireless Communications and Mobile Computing Conference (IWCMC)*, Istanbul, Turkey, 2011, pp. 2009–2014.
- [37] S. Sathish and C. di Flora, "Supporting smart space infrastructures: A dynamic context-model composition framework," in *Proceedings of the 3rd International Conference on Mobile Multimedia Communications*, Nafpaktos, Greece, 2007, pp. 1–6.
- [38] Y. Suo and Y. Shi, "Towards Initiative Smart Space model," in *Proceedings of Third International Conference on Pervasive Computing and Applications (ICPCA 2008)*, Alexandria, Egypt, 2008, pp. 747–752.
- [39] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," *Data Knowl. Eng.*, vol. 25, no. 1–2, pp. 161–197, Mar. 1998.

- [40] D. Zhang, H. Huang, C.-F. Lai, X. Liang, Q. Zou, and M. Guo, "Survey on context-awareness in ubiquitous media," *Multimed. Tools Appl.*, vol. 67, no. 1, pp. 179–211, Nov. 2013.
- [41] Resource Description Framework (RDF). Available: <https://www.w3.org/RDF/>.
- [42] RDF Schema. Available: <https://www.w3.org/TR/rdf-schema/>.
- [43] Web Ontology Language (OWL). Available: <https://www.w3.org/OWL/>.
- [44] F. Aiello, G. Fortino, R. Gravina, and A. Guerrieri, "A Java-Based Agent Platform for Programming Wireless Sensor Networks," *Comput. J.*, vol. 54, no. 3, pp. 439–454, Mar. 2011.
- [45] C. Nugent, D. Finlay, G. Moore, H. McDonald, J. Hallberg, and W. Burns, "Assessing the Impact of the homeML Format and the homeML Suite within the Research Community," *JUCS - J. Univers. Comput. Sci.*, no. 17, Nov. 2013.
- [46] B. Gonçalves and J. G. Pereira Filho, "An ECG Markup Language for Ambulatory Telemonitoring and Decision Making Support," in *Proceedings of the First International Conference on Health Informatics*, Funchal, Madeira, Portugal, pp. 37–43.
- [47] V. Rialle, J.-B. Lamy, N. Noury, and L. Bajolle, "Telemonitoring of patients at home: a software agent approach," *Comput. Methods Programs Biomed.*, vol. 72, no. 3, pp. 257–268, Nov. 2003.
- [48] K. Henriksen and I. Jadwiga, "A software engineering framework for context-aware pervasive computing," in *PERCOM '04 Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04)*, 2004.
- [49] A. Vlachostergiou, G. Stratogiannis, G. Caridakis, G. Siolas, and P. Mylonas, "User Adaptive and Context-Aware Smart Home Using Pervasive and Semantic Technologies," *J. Electr. Comput. Eng.*, vol. 2016, pp. 1–20, 2016.
- [50] B. Bouchard, S. Giroux, and A. Bouzouane, "A Smart Home Agent for Plan Recognition of Cognitively-impaired Patients," *J. Comput.*, vol. 1, no. 5, Aug. 2006.
- [51] L. Chen, C. Nugent, M. Mulvenna, D. Finlay, X. Hong, and M. Poland, "Using Event Calculus for Behaviour Reasoning and Assistance in a Smart Home," in *Smart Homes and Health Telematics*, vol. 5120, S. Helal, S. Mitra, J. Wong, C. K. Chang, and M. Mokhtari, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 81–89.
- [52] G. Okeyo, L. Chen, and H. Wang, "Combining ontological and temporal formalisms for composite activity modelling and recognition in smart homes," *Future Gener. Comput. Syst.*, vol. 39, pp. 29–43, Oct. 2014.
- [53] H. Chen, F. Perich, T. Finin, and A. Joshi, "SOUPA: standard ontology for ubiquitous and pervasive applications," in *Proceedings of the First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS 2004)*, Boston, MA, USA, 2004, pp. 258–267.
- [54] J. Ye, G. Stevenson, and S. Dobson, "USMART: An Unsupervised Semantic Mining Activity Recognition Technique," *ACM Trans. Interact. Intell. Syst.*, vol. 4, no. 4, pp. 1–27, Nov. 2014.
- [55] T. Lukasiewicz and U. Straccia, "Managing uncertainty and vagueness in description logics for the Semantic Web," *Web Semant. Sci. Serv. Agents World Wide Web*, vol. 6, no. 4, pp. 291–308, Nov. 2008.
- [56] G. Liang and J. Cao, "Social context-aware middleware: A survey," *Pervasive Mob. Comput.*, vol. 17, pp. 207–219, Feb. 2015.
- [57] W3C Uncertainty Reasoning for the World Wide Web Incubator Group. Available: <https://www.w3.org/2005/Incubator/urw3/>.
- [58] Uncertainty Ontology. Available: <https://www.w3.org/2005/Incubator/urw3/wiki/UncertaintyOntology.html>.
- [59] P. C. G. da Costa, K. B. Laskey, and K. J. Laskey, "PR-OWL: A Bayesian Ontology Language for the Semantic Web," in *Uncertainty Reasoning for the Semantic Web I*, vol.

- 5327, P. C. G. da Costa, C. d'Amato, N. Fanizzi, K. B. Laskey, K. J. Laskey, T. Lukasiewicz, M. Nickles, and M. Pool, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 88–107.
- [60] B. Andrea and T. Franco, "Extending ontology queries with Bayesian network reasoning," in *Proceedings of International Conference on Intelligent Engineering Systems (INES 2009)*, Barbados, Barbados, 2009, pp. 165–170.
 - [61] Y. Yang and J. Calmet, "OntoBayes: An Ontology-Driven Uncertainty Model," in *CIMCA '05 Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, Vienna, Austria, 2005, vol. 01, pp. 457–463.
 - [62] F. Bobillo and U. Straccia, "Fuzzy ontologies and fuzzy integrals," in *Proceedings of 11th International Conference on Intelligent Systems Design and Applications (ISDA)*, Cordoba, Spain, 2011, pp. 1311–1316.
 - [63] F. Bobillo and U. Straccia, "fuzzyDL: An expressive fuzzy description logic reasoner," in *Proceedings of IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2008)*, Hong Kong, China, 2008, pp. 923–930.
 - [64] N. D. Rodriguez, "Semantic and fuzzy modelling for human behaviour recognition in smart spaces: a case study on ambient assisted living." Ph.D. dissertation, the department of information technologies, Åbo Akademi University, 2015.
 - [65] X. Li, J.-F. Martinez, and G. Rubio, "A New Fuzzy Ontology Development Methodology (FODM) Proposal," *IEEE Access*, vol. 4, pp. 7111–7124, 2016.
 - [66] F. Bobillo and U. Straccia, "The fuzzy ontology reasoner? fuzzyDL," *Knowl.-Based Syst.*, vol. 95, pp. 12–34, Mar. 2016.
 - [67] R. N. Carvalho, "Probabilistic Ontology: Representation and Modeling Methodology," department of systems engineering and operations research, George Mason University, 2011.
 - [68] K. B. Laskey, "MEBN: A language for first-order Bayesian knowledge bases," *Artif. Intell.*, vol. 172, no. 2–3, pp. 140–178, Feb. 2008.
 - [69] Directed Acyclic Graph (DAG). Available: https://en.wikipedia.org/wiki/Directed_acyclic_graph.
 - [70] S. Matsumoto, R. N. Carvalho, M. Ladeira, P. Costa, L. Lima dos Santos, and D. Silva, "UnBBayes: a Java Framework for Probabilistic Models in AI," in *Java in Academia and Research*, iConcept Press, 2011.
 - [71] R. Carvalho, K. Laskey, P. Costa, M. Ladeira, L. Santos, and S. Matsumoto, "UnBBayes: Modeling Uncertainty for Plausible Reasoning in the Semantic Web," in *Semantic Web*, G. Wu, Ed. InTech, 2010.
 - [72] UnBBayes project. Available: <http://unbbayes.sourceforge.net/>.
 - [73] J. Hertzberg and A. Saffiotti, "Using semantic knowledge in robotics," *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 875–877, Nov. 2008.
 - [74] IEEE SA-ORA-Ontologies for Robotics and Automation. Available: <https://standards.ieee.org/develop/wg/ORA.html>.
 - [75] S. R. Fiorini *et al.*, "A Suite of Ontologies for Robotics and Automation [Industrial Activities]," *IEEE Robot. Autom. Mag.*, vol. 24, no. 1, pp. 8–11, Mar. 2017.
 - [76] C. C. Insaurralde, J. J. Cartwright, and Y. R. Petillot, "Cognitive Control Architecture for autonomous marine vehicles," in *Proceedings of 2012 IEEE International Systems Conference (SysCon)*, Vancouver, BC, Canada, 2012, pp. 1–8.
 - [77] M. Tenorth and M. Beetz, "Representations for robot knowledge in the KnowRob framework," *Artif. Intell.*, vol. 247, pp. 151–169, Jun. 2017.
 - [78] S. Lemaignan, R. Ros, L. Mösenlechner, R. Alami, and M. Beetz, "ORO, a knowledge management platform for cognitive architectures in robotics," in *Proceedings of 2010*

- IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, 2010, pp. 3548–3553.
- [79] S. Dhouib, S. Kchir, S. Stinckwich, T. Ziadi, and M. Ziane, “RobotML, a Domain-Specific Language to Design, Simulate and Deploy Robotic Applications,” presented at the Third international conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR’12), Tsukuba, Japan, 2012, vol. 7628, pp. 149–160.
 - [80] P. Martinet and B. Patin, “PROTEUS: A platform to organise transfer inside French robotic community,” presented at the 3rd National Conference on “Control Architectures of Robots,” Bourges, France, 2008.
 - [81] The PROTEUS Project. Available: <http://www.anr-proteus.fr/>.
 - [82] V. I. Gorodetsky, V. V. Samoylov, and D. V. Trotskii, “The reference ontology of collective behavior of autonomous agents and its extensions,” *J. Comput. Syst. Sci. Int.*, vol. 54, no. 5, pp. 765–782, Sep. 2015.
 - [83] A. Bouguerra, L. Karlsson, and A. Saffiotti, “Monitoring the execution of robot plans using semantic knowledge,” *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 942–954, Nov. 2008.
 - [84] C. Galindo, J.-A. Fernández-Madrigal, J. González, and A. Saffiotti, “Robot task planning using semantic maps,” *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 955–966, Nov. 2008.
 - [85] C. Schlenoff, S. Balakirsky, M. Uschold, R. Provine, and S. Smith, “Using ontologies to aid navigation planning in autonomous vehicles,” *Knowl. Eng. Rev.*, vol. 18, no. 3, pp. 243–255, Sep. 2003.
 - [86] C. Schlenoff and E. Messina, “A robot ontology for urban search and rescue,” in *Proceedings of the 2005 CIKM Conference: Workshop on Research in Knowledge Representation for Autonomous Systems*, Bremen, Germany, 2005, pp. 27–34.
 - [87] K. J. Witt, J. Stanley, and D. Smithbauer, “Enabling Sensor Webs by Utilizing SWAMO for Autonomous Operations,” in *KAR’05 Proceedings of the 2005 ACM workshop on Research in knowledge representation for autonomous systems*, Bremen, Germany, 2005, pp. 27–34.
 - [88] E. Miguelanez, P. Patron, K. E. Brown, Y. R. Petillot, and D. M. Lane, “Semantic Knowledge-Based Framework to Improve the Situation Awareness of Autonomous Underwater Vehicles,” *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 5, pp. 759–773, May 2011.
 - [89] C. Bettini *et al.*, “A survey of context modelling and reasoning techniques,” *Pervasive Mob. Comput.*, vol. 6, no. 2, pp. 161–180, Apr. 2010.
 - [90] K.E. Ko and K.B. Sim, “Development of context aware system based on Bayesian network driven context reasoning method and ontology context modeling,” in *Proceedings of International Conference on Control, Automation and Systems (ICCAS 2008)*, Seoul, South Korea, 2008, pp. 2309–2313.
 - [91] G. Pilato, A. Augello, M. Missikoff, and F. Taglino, “Integration of Ontologies and Bayesian Networks for Maritime Situation Awareness,” in *Proceedings of 2012 IEEE Sixth International Conference on Semantic Computing (ICSC)*, Palermo, Italy, 2012, pp. 170–177.
 - [92] S. Rho, S. Song, E. Hwang, and M. Kim, “COMUS: Ontological and Rule-Based Reasoning for Music Recommendation System,” in *Advances in Knowledge Discovery and Data Mining*, vol. 5476, T. Theeramunkong, B. Kijssirikul, N. Cercone, and T.-B. Ho, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 859–866.
 - [93] R. N. Carvalho, S. Matsumoto, K. B. Laskey, P. C. G. Costa, M. Ladeira, and L. L. Santos, “Probabilistic Ontology and Knowledge Fusion for Procurement Fraud Detection in Brazil,” in *Uncertainty Reasoning for the Semantic Web II*, vol. 7123, F. Bobillo, P. C. G. Costa, C. d’Amato, N. Fanizzi, K. B. Laskey, K. J. Laskey, T. Lukasiewicz, M. Nickles, and M. Pool, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 19–40.

- [94] Y. Fischer and J. Beyerer, "Ontologies for probabilistic situation assessment in the maritime domain," in *Proceedings of 2013 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, San Diego, CA, USA, 2013, pp. 102–105.
- [95] G. Chantas *et al.*, "Multi-Entity Bayesian Networks for Knowledge-Driven Analysis of ICH Content," in *Computer Vision - ECCV 2014 Workshops*, vol. 8926, L. Agapito, M. M. Bronstein, and C. Rother, Eds. Cham: Springer International Publishing, 2015, pp. 355–369.
- [96] S. Raza, S. Haider, and M.-A. Williams, "Robot Reasoning Using First Order Bayesian Networks," in *Integrated Uncertainty in Knowledge Modelling and Decision Making*, vol. 8032, Z. Qin and V.-N. Huynh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 1–12.
- [97] SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Available: <https://www.w3.org/Submission/SWRL/>.
- [98] OODA (Observe, Oriented, Descision, and Act). Available: <http://www.dtic.mil/dtic/tr/fulltext/u2/a590672.pdf>.
- [99] H. Pung *et al.*, "Context-aware middleware for pervasive elderly homecare," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 510–524, May 2009.
- [100] P. Hu, S. K. Chong, J. Indulska, and S. Krishnaswamy, "Context-aware and resource efficient sensing infrastructure for context-aware applications," in *Proceedings of 2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, Mannheim, Germany, 2010, pp. 492–498.
- [101] S. K. Chong, M. M. Gaber, S. W. Loke, and S. Krishnaswamy, "ARTS: Adaptive Rule Triggers on Sensors for Energy Conservation in Applications using Coarse-Granularity Data," in *Proceedings of International Conference on Embedded Software and Systems (ICESS'08)*, Chengdu, Sichuan, China, 2008, pp. 314–321.
- [102] The FIWARE Project. Available: <https://www.fiware.org/>.
- [103] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "CA4IOT: Context Awareness for Internet of Things," in *Proceedings of the 2012 IEEE International Conference on Green Computing and Communications (GREENCOM'12)*, Besancon, France, 2012, pp. 775–782.
- [104] E. J. Y. Wei and A. T. S. Chan, "CAMPUS: A middleware for automated context-aware adaptation decision making at run time," *Pervasive Mob. Comput.*, vol. 9, no. 1, pp. 35–56, Feb. 2013.
- [105] J. K. - and S. P. -, "Context-Aware Services Framework Based on Semantic Web Services for Automatic Discovery and Integration of Context," *Int. J. Adv. Comput. Technol.*, vol. 5, no. 4, pp. 439–448, Feb. 2013.
- [106] OWL-S: Semantic Markup for Web Services. Available: <https://www.w3.org/Submission/OWL-S/>.
- [107] SOAP (Simple Object Access Protocol). Available: <https://www.w3.org/TR/soap/>.
- [108] A. Huertas Celdran, F. J. Garcia Clemente, M. Gil Perez, and G. Martinez Perez, "SeCoMan: A Semantic-Aware Policy Framework for Developing Privacy-Preserving and Context-Aware Smart Applications," *IEEE Syst. J.*, vol. 10, no. 3, pp. 1111–1124, Sep. 2016.
- [109] A. Forkan, I. Khalil, and Z. Tari, "CoCaMAAL: A cloud-oriented context-aware middleware in ambient assisted living," *Future Gener. Comput. Syst.*, vol. 35, pp. 114–127, Jun. 2014.
- [110] A. Forkan, I. Khalil, A. Ibaida, and Z. Tari, "BDCaM: Big Data for Context-aware Monitoring - A Personalized Knowledge Discovery Framework for Assisted Healthcare," *IEEE Trans. Cloud Comput.*, pp. 1–1, 2015.

- [111] G. Wu, H. Zhang, M. Qiu, Z. Ming, J. Li, and X. Qin, "A decentralized approach for mining event correlations in distributed system monitoring," *J. Parallel Distrib. Comput.*, vol. 73, no. 3, pp. 330–340, Mar. 2013.
- [112] M. A. E. Khaddar, M. Chraibi, H. Harroud, and M. Boulmalf, "A policy-based middleware for context-aware pervasive computing," *International Journal of Pervasive Computing and Communications*, vol. 11, pp. 43–68, 2015.
- [113] F. Maurelli, Z. Saigol, and G. Papadimitriou, "Probabilistic approaches in ontologies: Joining semantics and uncertainty for AUV persistent autonomy," presented at the Oceans - San Diego, San Diego, CA, USA, 2013.
- [114] M. C. Suárez-Figueroa, A. Gómez-Pérez, and M. Fernando López, "The NeOn Methodology framework: A scenario-based methodology for ontology development," *Appl. Ontol.*, vol. 10, no. 2, pp. 107–145, Sep. 2015.
- [115] M. Fernandez, A. Gomez-Perez, and N. Juristo, "METHONTOLOGY: From Ontological Art Towards Ontological Engineering," AAI Technical Report SS-97-06, 1997.
- [116] N. F. Noy and D. L. McGuinness, "Ontology development 101: A guide to creating your first ontology," Stanford Medical Informatics, 2001.
- [117] D. Vrandečić, S. Pinto, C. Tempich, and Y. Sure, "The DILIGENT knowledge processes," *J. Knowl. Manag.*, vol. 9, no. 5, pp. 85–96, Oct. 2005.
- [118] Y. Sure, S. Staab, and R. Studer, "On-To-Knowledge Methodology (OTKM)," in *Handbook on Ontologies*, S. Staab and R. Studer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 117–132.
- [119] K. Kotis and G. A. Vouros, "Human-centered ontology engineering: The HCOME methodology," *Knowl. Inf. Syst.*, vol. 10, no. 1, pp. 109–131, Jul. 2006.
- [120] M. Jarrar and R. Meersman, "Ontology Engineering ? The DOGMA Approach," in *Advances in Web Semantics I*, vol. 4891, T. S. Dillon, E. Chang, R. Meersman, and K. Sycara, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 7–34.
- [121] M. Fernandez-Lopez and A. Gomez-Perez, "Overview and analysis of methodologies for building ontologies," *Knowl. Eng. Rev.*, vol. 17, no. 02, Jun. 2002.
- [122] A. Sheth and M. D. Lytras, Eds., *Semantic Web-Based Information Systems: State-of-the-Art Applications*. IGI Global, 2007.
- [123] A. Zouaq and R. Nkambou, "A Survey of Domain Ontology Engineering: Methods and Tools," in *Advances in Intelligent Tutoring Systems*, vol. 308, R. Nkambou, J. Bourdeau, and R. Mizoguchi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 103–119.
- [124] P. Alexopoulos, M. Wallace, K. Kafentzis, and D. Askounis, "IKARUS-Onto: a methodology to develop fuzzy ontologies from crisp ones," *Knowl. Inf. Syst.*, vol. 32, no. 3, pp. 667–695, Sep. 2012.
- [125] H. Ghorbel, A. Bahri, and R. Bouaziz, "Fuzzy ontologies building method: Fuzzy OntoMethodology," in *Proceedings of 2010 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS)*, Toronto, ON, Canada, 2010, pp. 1–8.
- [126] R. N. Carvalho, K. B. Laskey, and P. C. G. D. Costa, "Uncertainty modeling process for semantic technology," *PeerJ Comput. Sci.*, vol. 2, p. e77, Aug. 2016.
- [127] W3C wiki. Available: https://www.w3.org/wiki/Main_Page.
- [128] Swoogle. Available: <http://swoogle.umbc.edu/2006/>.
- [129] U. Straccia, *Foundations of Fuzzy Logic and Semantic Web Languages*. Chapman & Hall/CRC, 2013.
- [130] U. Straccia, "A Minimal Deductive System for General Fuzzy RDF," in *Web Reasoning and Rule Systems*, vol. 5837, A. Polleres and T. Swift, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 166–181.
- [131] D. H. Fudholi, N. Maneerat, R. Varakulsiripunth, and Y. Kato, "Application of Protege, SWRL and SQWRL in fuzzy ontology-based menu recommendation," in *Proceedings of*

- International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2009)*, Kanazawa, Japan, 2009, pp. 631–634.
- [132] S. Borgwardt, F. Distel, and R. Peñaloza, “The limits of decidability in fuzzy description logics with general concept inclusions,” *Artif. Intell.*, vol. 218, pp. 23–55, Jan. 2015.
 - [133] F. Bobillo, M. Delgado, and J. Gómez-Romero, “DeLorean: A reasoner for fuzzy OWL 2,” *Expert Syst. Appl.*, vol. 39, no. 1, pp. 258–272, Jan. 2012.
 - [134] I. Jacobson, G. Booch, and J. Rumbaugh, *The unified software development process*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.
 - [135] *Ontological Engineering*. London: Springer-Verlag, 2004.
 - [136] PROTÉGÉ. Available: <http://protege.stanford.edu/>.
 - [137] Pellet reasoner. Available: <https://github.com/stardog-union/pellet>.
 - [138] Simple Vehicle Identification MTheory: CogSIMA 2016 Tutorial. Available: ftp://197.155.77.5/sourceforge/u/un/unbbayes/UnBBayes%20Plugin%20Framework/Plugins/Probabilistic%20Networks/MEBN/1.14.X/Tutorial_MEBN_SimpleVehicleID.pdf.
 - [139] A. Zagorecki, “Local Probability Distributions in Bayesian Networks: Knowledge Elicitation and Inference,” University of Pittsburgh, Pittsburgh, PA, USA, 2010.
 - [140] K. B. Laskey and S. M. Mahoney, “Network engineering for agile belief network models,” *IEEE Trans. Knowl. Data Eng.*, vol. 12, no. 4, pp. 487–498, Aug. 2000.
 - [141] A. Sánchez-Macián, E. Pastor, J. E. de López Vergara, and D. López, “Extending SWRL to Enhance Mathematical Support,” in *Web Reasoning and Rule Systems*, vol. 4524, M. Marchiori, J. Z. Pan, and C. de S. Marie, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 358–360.
 - [142] L. Christoph, “Ontologies and languages for representing mathematical knowledge on the Semantic Web,” *Semantic Web*, no. 2, pp. 119–158, 2013.
 - [143] MathML. Available: <https://www.w3.org/Math/whatIsMathML.html>.
 - [144] OpenMath. Available: <https://www.openmath.org/>.
 - [145] X. Li, J.-F. Martinez, M. Eckert, and G. Rubio, “Uncertainty Quantification in Mathematics-Embedded Ontologies Using Stochastic Reduced Order Model,” *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 912–920, Apr. 2017.
 - [146] M. Grigoriu, “Reduced order models for random functions. Application to stochastic problems,” *Appl. Math. Model.*, vol. 33, no. 1, pp. 161–175, Jan. 2009.
 - [147] J. E. Warner, M. Grigoriu, and W. Aquino, “Stochastic reduced order models for random vectors: Application to random eigenvalue problems,” *Probabilistic Eng. Mech.*, vol. 31, pp. 1–11, Jan. 2013.
 - [148] Q. Du, V. Faber, and M. Gunzburger, “Centroidal Voronoi Tessellations: Applications and Algorithms,” *SIAM Rev.*, vol. 41, no. 4, pp. 637–676, Jan. 1999.
 - [149] J. E. Warner, W. Aquino, and M. D. Grigoriu, “Stochastic reduced order models for inverse problems under uncertainty,” *Comput. Methods Appl. Mech. Eng.*, vol. 285, pp. 488–514, Mar. 2015.
 - [150] García de Prado Ortiz and G. Ortiz, “Context-aware services: A survey on current proposals,” presented at the Third International Conferences on Advanced Service Computing, 2011.
 - [151] Cumulative Distribution Function (CDF). Available: https://en.wikipedia.org/wiki/Cumulative_distribution_function.
 - [152] The OWL entailment. Available: <https://www.w3.org/TR/owl2-profiles/>.
 - [153] The HerMiT reasoner. Available: <http://www.hermit-reasoner.com/>.
 - [154] The FaCT++ reasoner. Available: <http://owl.man.ac.uk/factplusplus/>.
 - [155] The Rule Markup Language (RuleML). Available: http://wiki.ruleml.org/index.php/RuleML_Home.
 - [156] Horn-like rules. Available: <https://cs.nyu.edu/courses/spring02/G22.2560-001/horn.html>.

- [157] PowerLoom® Knowledge Representation & Reasoning System. Available: <https://www.isi.edu/isd/LOOM/PowerLoom/>.
- [158] Jena OWL reasoner. Available: <https://jena.apache.org/documentation/inference/>.
- [159] Jena Ontology API. Available: <https://jena.apache.org/documentation/ontology/>.
- [160] Application Programming Interfaces (APIs). Available: https://en.wikipedia.org/wiki/Application_programming_interface.
- [161] URI (Uniform Resource Identifier). Available: https://en.wikipedia.org/wiki/Uniform_Resource_Identifier.
- [162] SPARQL (SPARQL Protocol and RDF Query Language). Available: <https://www.w3.org/TR/rdf-sparql-query/>.
- [163] Universal Binary Format (UBF). Available: <http://ubf.github.io/ubf/ubf-user-guide.en.html>.
- [164] P. Murillo Dávila, "Hybrid knowledge representation and reasoning in a context-awareness component for underwater cognitive and cooperative vehicles," Bachelor thesis, Universidad Politécnica de Madrid, Madrid, Spain, 2017.