Digital Ecosystem: For Better Management of Multimedia Contents

Solomon Asres Kidanu

Université de Pau et des Pays de l'Adour (UPPA)
Ecole Doctorale des Sciences Exactes et Leurs Applications
LABORATOIRE INFORMATIQUE - LIUPPA

Digital Ecosystem: For Better Management of Multimedia Contents

(Écosystémes numériques: pour une meilleure gestion des contenus multimédias)

par

Solomon Asres Kidanu

THÉSE

Pour obtenir le grade de **DOCTEUR** de l'**UPPA**

Spécialité : **Informatique**

Soutenance le 09 décembre 2016

Devant le jury composé de

Djamal BENSLIMANE, Professeur (Univeristé Claude Bernard Lyon 1), Rapporteur
Kokou YETONGNON, Professeur (Université de Bourgogne), Rapporteur
Ernesto EXPOSITO, Professeur (Université de Pau et des Pays de l'Adour), Examinateur
Marta RUKOZ, Professeur (Université Paris Ouest Nanterre La Défense), Examinateur
Richard CHBEIR, Professeur (Université de Pau et des Pays de l'Adour), Directeur de thése
Yudith CARDINALE, Professeur (Universidad Simón Bolívar), Co-directeur de thése

To **my grandmother**

and
my wife

"Nothing is impossible, the word itself says 'I'm possible'!"

Audrey Hepburn

ACKNOWLEDGEMENT

First and for most I would like to praise the Almighty God for giving me the power and capability to the work presented in this thesis. If it was not for His grace I would never have accomplished any of this. Undertaking this PhD has been a truly challenging experience for me and through it all I have received guidance and support from many people, in which I am very grateful.

My very special hearfelt thanks goes to my principal advisor **Prof. Richard Chbeir** for accepting me as a PhD student, for his continuous support, guidance, and critical comments. Similar profound gratitude goes to my co-advisor **Prof. Yudith Cardinale**. I am particularly indebted for her constant faith in my work, unlimited support, and especially for her patience and guidance during my writing process. I would also like to thank Gilbert Tekli for his support; and my appreciation to Corentine Donzeil for his remarkable assistance in the prototype development of my work.

I would like to send my sincere thanks to the French Embassy in Ethiopia for their financial support and especially for Ms. Bethel Zemedekun for her unlimited support in facilitating the scholarship. I would also like to thank the Department of Computer Science, Addis Ababa University for giving me the chance to pursue this study. My special thank to my friend Ashenafi Kassahun for taking care of all my stuffs in Ethiopia on behalf of me in all these years.

I am very thankful and would like to thank the thesis committee: Prof. Kouku Yetongnon and Prof. Djamal Benslimane for their willingness to be rapporteurs for my thesis; Prof. Marta Rukoz and Prof. Ernesto Exposito for being part of my PhD examination committee.

I am also indebted to thank all my friends and colleagues in the lab and specially to Regina, Riadh, Khouloude, Eliana, Ghada, Chinnapong, Irvin; and staff members of IUT de Bayonne et du Pays Basque who were always so helpful in numerous ways.

And here I am today, with the prayers, support, and encouragement of fathers, brothers, and sisters in **Mahibere Kidusan** which I love and serve dearly. Exceptional thanks go to the family of Kassahun H/Mariam, Dr. Semu Mitiku, Tesfaye Bihonegn, and Dn. Birhanu Admas.

I am very thankful for all my family members and more importantly and finally to my better half, Yodit Ayalew. Her encouragement, quiet patience, tolerance, and unwavering love were always my inspiration. Words cannot express how grateful I am to my beloved wife to her support in every possible way to see the completion of this work.

Résumé

De nos jours, la plupart des informations échangées sur Internet (notamment via les réseaux sociaux) se présente sous forme de données multimédias. Chaque acteur sur Internet (particuliers, entreprises, communautés territoriales, etc.) devient, à la fois, un producteur et un consommateur de données. La croissance de l'offre et de la demande des données multimédias permet aux producteurs / consommateurs d'avoir plus de choix et de possibilités de partage des données dans des environnements collaboratifs. Néanmoins, les plates-formes des réseaux sociaux et d'autres environnements collaboratifs traditionnels présentent des limites concernant l'enrichissement, l'extraction de la sémantique et la combinaison des ressources multimédias de différentes sources ne permettant pas ainsi de faire émerger une intelligence collective pour une meilleure recommandation, négociation, notification, etc. En outre, vu l'augmentation des ressources multimédias dans le web, les plates-formes existantes souffrent de plusieurs difficultés à conserver les avantages mutuels des acteurs dans un schéma gagnant-gagnant. En particulier, elles ne permettent pas aux utilisateurs de définir leurs profils et leurs préférences, de publier ou de conserver localement leurs ressources, d'avoir le contrôle sur leurs ressources selon leurs propres règles d'utilisation et d'usage. Le maintien de l'équilibre entre les ressources et la consommation est un autre défi pour assurer la survie et la fiabilité des environnements collaboratifs.

Dans ce contexte, les écosystèmes numériques ont émergé depuis une décennie en proposant un environnement numérique permettant à ses participants de maintenir une coopération plus harmonieuse et qui favorise davantage l'extraction et la promotion d'une connaissance collective. Dans cette thèse, nous adoptons ce concept d'écosystèmes numériques pour fournir une meilleure gestion des contenus multimédias assurant les bénéfices de tous ses participants. Les écosystèmes numériques sont souvent décrits comme des systèmes complexes dans lesquels plusieurs entités existent et interagissent. Pour modéliser et développer de tels systèmes complexes, plusieurs langages de modélisation sont nécessaires. Cependant, dans le contexte des écosystèmes numériques, il est toujours nécessaire de disposer d'un langage et d'un framework de modélisation complets pour représenter les entités nécessaire, faciliter le processus de développement et réduire sa complexité.

La première contribution de ce travail consiste à proposer une modélisation des écosystèmes numériques. Nous proposons un modèle ontologique fondé sur des concepts des Systèmes Multi-Agents (SMA), appelé MAS2DES-Onto. Ce dernier est constitué de cinq modules pour représenter tous les aspects essentiels des agents dans le contexte des écosystèmes numériques : Structure, Espèces, Raisonnement, Interaction et Système. La deuxième contribution de cette thèse est liée au processus de développement des écosystèmes

numériques. Nous proposons un framework appelé Onto2MAS pour le développement facile et rapide des écosystèmes numériques basés surMAS2DES-Onto. Le framework comporte trois composants (Designer, Generator et Deployer) pour prendre en compte les processus de conception, de génération et de déploiement du développement. Onto2MAS fournit également un langage, appelée OJ, qui est un langage simplifié pour aider le développeur à spécifier ses besoins. Pour valider notre approche, nous présentons également les résultats des tests expérimentaux que nous avons menés avec une première implémentation de Onto2MAS, appelée OnToJade. L'outil est développé avec des plate-formes connues, telles que JADE, Jena, Java et Protégé. Enfin, nous présentons un écosystème numérique multimédia (MMDES) comme un nouvel environnement de collaboration et de partage des contenus multimédias générés par MAS2DES-Onto et utilisant le framework Onto2MAS. Nous montrons comment les exigences particulières du MMDES (telles que la gestion des ressources multimédias, la gestion des connaissances et des requêtes, la contribution des participants aux ressources multimédias et l'équilibre de l'écosystème) sont traitées. La première version de l'implémentation MMDES est déployée sur une plate-forme mobile.

Mots-clés : Écosystèmes numériques, Multimédia, Systèmes multi-agents, Ontologie

Abstract

Nowadays, most of the information exchanged in Internet (particularly through social networks) is in the form of multimedia data. Each actor in Internet (individuals, enterprises, territorial communities, etc.) becomes producer and consumer of contents. Development and increasing demand for multimedia data offer producers/consumers more choices and opportunities for sharing in collaborative environments. Nevertheless, social network platforms and other traditional collaborative environments present limitations regarding to enrich, extract semantics, and combine multimedia resources from different sources so to come up with a collective intelligence for effective and better recommendation, negotiation, notification, etc. Existing collaborative environments also have constraints in addressing the extremely increasing multimedia resources in the web by keeping mutual benefits in a win-win situation. They do not fully allow users to define their profiles and preferences, to publish or keep locally their resources, to have control over their resources according to predefined usage rules, and to have the benefits from collective knowledge. Keeping balance in terms of resource provision and consumption is another challenge to ensure the survival and reliability of the collaborative environments.

In this context, Digital Ecosystems aim at creating a digital environment for interested participants that supports in-between cooperation and promotes collective knowledge sharing in order to provide mutual benefits, as a new way to handle collaboration in a distributed and heterogeneous environment. Thus, we propose a Digital Ecosystem for better management of multimedia contents ensuring the benefits of all its participants. The objective is to create a digital environment for interested participants that support in-between cooperation and promote collective knowledge sharing in order to provide mutual benefits, as a new way to handle collaboration in a distributed and heterogeneous environment. Digital Ecosystems are often described as complex systems in which several entities exist and interact. To model and develop such complex systems, modeling languages and frameworks are required. Multi-Agent Systems (MASs) have received much attention in recent years because of their advantages on modeling complex distributed systems. However, in the context of Digital Ecosystems, there is still a need for comprehensive modeling language and framework to represent the entities and also facilitate the development process. Moreover, the development of MAS-based Digital Ecosystems remains a complicated task, which demands time and special programming skills. Thus, there is a need for proposing modeling languages and methodologies that support easy and quick way of development and also reduce the overall complexity of developing process.

The first contribution of this work regards to Digital Ecosystems modeling. We propose an ontological model based on MAS concepts, called MAS2DES-Onto, which meets the requirements of MASs and Digital Ecosystems. It provides a clear representation of agent concepts and relationships to support the modeling of agents' behavior, knowledge, rule, etc. MAS2DES-Onto consists of five modules to represent all the essential aspects of agents in the context of MAS-based Digital Ecosystems: Structure, Species, Reasoning, Interaction, and System. The second contribution is related to the development process of Digital Ecosystems. We propose a framework, called Onto2MAS, for easy and quick development of MAs-based Digital Ecosystems. This framework enables developers an automatic and rapid generation of MAs-based Digital Ecosystems, based on the ontological model. The framework has three components (Designer, Generator, and Deployer) to support the designing, generating, and deployment processes of the development. Onto2MAS also provides a language, called OJ, which is a simplified language to help the developer in the process of specifying end-user requirements. To demonstrate the efficiency of our approach, we also present the results of experimental tests that we conducted with a first implementation of Onto2MAS, called OnToJade. It is developed with well-known platforms, such as JADE, Jena, Java, and Protégé Editor. Finally, we provide a MultiMedia Digital Ecosystem (MMDES) as a new environment of collaboration and sharing of multimedia contents generated from MAS2DES-Onto and using Onto2MAS framework. We show how particular requirements of MMDES (such as multimedia resources management, knowledge and query management, contribution of participants for multimedia resources, and ensuring balance of the ecosystem) are handled. The first version of MMDES implementation is deployed on a mobile platform.

Keywords:- Digital Ecosystems, Multimedia, Multi-Agent Systems, Ontology

Contents

Abstract vii

List of Figures xiii

List of Tables xv

1 Introduction 1
- 1.1 Context . . . 1
- 1.2 Motivating Scenario and Challenges . . . 3
- 1.3 Contribution of this study . . . 5
- 1.4 Publications . . . 6
- 1.5 Organization of the Thesis . . . 7

2 Related Work 9
- 2.1 Introduction . . . 9
- 2.2 Concepts of Digital Ecosystems . . . 11
  - 2.2.1 Overview . . . 11
  - 2.2.2 Components of Digital Ecosystems . . . 13
    - 2.2.2.1 Species . . . 14
    - 2.2.2.2 Environment . . . 14
  - 2.2.3 Characteristics of Digital Ecosystems . . . 15
  - 2.2.4 Categories of Digital Ecosystems . . . 17
    - 2.2.4.1 Digital Business Ecosystem . . . 17
    - 2.2.4.2 Digital Knowledge Ecosystem . . . 18
    - 2.2.4.3 Digital Service Ecosystem . . . 19
    - 2.2.4.4 Discussion . . . 19
- 2.3 Frameworks and Architectures of Digital Ecosystems . . . 20
  - 2.3.1 Dynamic Agent-based Ecosystem Model (DAEM) . . . 20
  - 2.3.2 Framework for Business Ecosystem Analysis and Modeling (BEAM) 21
  - 2.3.3 Digital Business Ecosystem (DBE) Framework . . . 21
  - 2.3.4 Service-Oriented Peer-to-Peer Architecture for Digital Ecosystems 22
  - 2.3.5 Ecosystem-Oriented Architecture . . . 22
  - 2.3.6 Modeling Framework for Service Ecosystems . . . 23
  - 2.3.7 Discussion . . . 24
- 2.4 Ontology-Based MAS Development Methodologies . . . 25
  - 2.4.1 Introduction . . . 25
  - 2.4.2 MAS-CommonKADS Methodology . . . 26
  - 2.4.3 MESSAGE Methodology . . . 26

2.4.4 INGENIAS Methodology . . . 27
2.4.5 MaSE Methodology . . . 28
2.4.6 PASSI Methodology . . . 28
2.4.7 MOMA Methodology . . . 29
2.4.8 Discussion . . . 29
2.5 Agent Concept Representations . . . 30
2.5.1 Introduction . . . 30
2.5.2 Agent Concept Representations in MASs . . . 32
2.5.2.1 Conceptual Agent Model . . . 32
2.5.2.2 Architecture Description Language Concepts for Agent . . . 34
2.5.2.3 Agent Model for Ontology-Driven Procedural Reasoning System . . . 35
2.5.2.4 Emotional Belief-Desire-Intention Model . . . 36
2.5.2.5 ANote Model . . . 37
2.5.2.6 MESSAGE Concepts . . . 38
2.5.2.7 INGENIAS Model . . . 39
2.5.2.8 Agent concept in Ontology-based Modelling of MAS . . . 39
2.5.3 Agent Concept Representation in Digital Ecosystems . . . 39
2.5.4 Discussion . . . 40
2.6 Conclusion . . . 42

3 Ontology for MAS-Based Digital Ecosystems 43
3.1 Introduction . . . 43
3.2 Modeling Requirements for Multi-Agent Systems and Digital Ecosystems 45
3.2.1 MAS Modeling Requirements . . . 45
3.2.2 Digital Ecosystem Requirements . . . 46
3.3 MAS-based Digital Ecosystem for Documentary Movie Production . . . 47
3.4 MAS2DES-Onto . . . 50
3.4.1 Structural Module . . . 52
3.4.2 Species Module . . . 52
3.4.3 Reasoning Module . . . 54
3.4.4 Interaction Module . . . 56
3.4.5 System Module . . . 59
3.4.6 MAS2DES-Onto Relationships . . . 59
3.5 Conclusion . . . 62

4 Onto2MAS: Ontology-based Framework for MAS-Based Digital Ecosystems Generation 64
4.1 Introduction . . . 64
4.2 Onto2MAS Framework . . . 65
4.2.1 Onto2MAS Components . . . 66
4.2.1.1 Designer . . . 66
4.2.1.2 Generator . . . 67
4.2.1.3 Deployer . . . 68
4.2.2 OJ Language . . . 68
4.2.2.1 Conditional Statement . . . 68
4.2.2.2 Commands . . . 69

4.2.2.3 Basic Arithmetical Operations . . . 71
4.2.2.4 Mathematical Expressions . . . 71
4.3 OnToJade: Implementation of Onto2MAS . . . 73
4.3.1 Designer Component of OnToJade . . . 73
4.3.1.1 Derived Ontology . . . 74
4.3.1.2 Creating Ontology Representation Files . . . 78
4.3.2 Generator Component of OnToJade . . . 81
4.3.2.1 OWL Parser . . . 83
4.3.2.2 Agent Factory . . . 84
4.3.2.3 OJ Language Engine . . . 86
4.3.2.4 Agent Instantiator . . . 86
4.3.2.5 Visualizer . . . 89
4.4 Experimental Tests and Results . . . 89
4.4.1 Performance Test . . . 89
4.4.2 Agent Interaction Test . . . 91
4.4.3 Test for Comply System Level Rules . . . 96
4.5 Conclusion . . . 99

5 MMDES: MultiMedia Digital Ecosystem 100
5.1 Social Media and Collaborative Platforms . . . 101
5.1.1 Challenges of Social Media Platforms . . . 102
5.1.2 Distributed and Collaborative Environments . . . 104
5.2 MMDES: New Platform for Collaboration and Multimedia Sharing . . . 106
5.2.1 Overview of MMDES . . . 106
5.2.2 Objectives of MMDES . . . 108
5.3 Design Process of MMDES . . . 110
5.3.1 MMDES Derived Ontology . . . 110
5.3.2 MMDES Ontology File . . . 118
5.4 MMDES Agents Generation . . . 118
5.4.1 General Agents . . . 119
5.4.2 Coordinator Agents in MMDES . . . 119
5.4.3 Knowledge and Queries Management in MMDES . . . 121
5.4.4 Ensuring Equilibrium in MMDES . . . 123
5.4.4.1 Quantifiable and Sharable Resources . . . 123
5.4.4.2 Involved Agents for Ensuring Equilibrium . . . 125
5.5 Deployment of MMDES for a Mobile Platform . . . 127
5.5.1 User Registration Process . . . 129
5.5.2 Multimedia Content Queries . . . 129
5.6 Conclusion . . . 132

6 Conclusion and Future Work 133
6.1 Discussion . . . 133
6.2 Contribution . . . 135
6.3 Future work . . . 135

A Appendix A: Console outputs 138

A.1 Multicast Communication 138
A.2 Broadcast Communication 139
A.3 System Level Rules 141

B Appendix B: Résumé-Extended **142**
B.1 Contexte 142
B.2 Scénario de motivation et défis 144
B.3 Contribution de la thèse 147
B.4 Résumé des chapitres 148
B.5 Travaux futurs 150
B.6 Conclusion 152

Bibliography **153**

List of Figures

2.1 CAM Model [1] . . . 34
2.2 ADL Model [2] . . . 35
2.3 O-PRS Model [3] . . . 36
2.4 ANote Model [4] . . . 37
2.5 MESSAGE Model [5] . . . 38
2.6 INGENIAS Model Views [6] . . . 39
2.7 Main Concepts of MAS ontology model [7] . . . 40
2.8 Species Ontology of Digital Ecosystems [8] . . . 40

3.1 Conceptual Model for MAS-based Digital Ecosystem . . . 51
3.2 Structural Module of MAS2DES-Onto . . . 53
3.3 Species Module of MAS2DES-Onto . . . 54
3.4 Reasoning Module of MAS2DES-Onto . . . 56
3.5 Interaction Module of MAS2DES-Onto . . . 58
3.6 System Module of MAS2DES-Onto . . . 60
3.7 Detailed Conceptual Model for MAS-based Digital Ecosystem . . . 62

4.1 Onto2MAS Framework Components . . . 66
4.2 MAS2DES-Onto Conceptual Model . . . 67
4.3 Send Command Syntax . . . 69
4.4 Execution Command Syntax . . . 71
4.5 Architecture of OnToJade Prototype . . . 73
4.6 Derived Ontology for OnToJade . . . 75
4.7 Protégé Ontology Tool Interface . . . 78
4.8 Derived Ontology Classes . . . 79
4.9 Derived Ontology Top Data Property . . . 80
4.10 Behavior Property Sub-properties Description . . . 80
4.11 OnToJade Individuals by Type . . . 81
4.12 OnToJade Individual Agent Description . . . 82
4.13 OnToJade Agent Property Description . . . 82
4.14 Behavior Description using OJ Language . . . 83
4.15 Generator Component of OnToJade . . . 83
4.16 Unicast communication in DES1 . . . 93
4.17 Multicast communication in DES2 . . . 95
4.18 Broadcast communication in DES3 . . . 96
4.19 Test Result for Rule 2 . . . 98
4.20 Test Result for TimeToQuit . . . 99

5.1 MMDES Environment 107
5.2 MMDES Concepts 110
5.3 Classes and Instances of MMDES 118
5.4 Ranker Agent prioritize resources to Recommendation Agents 128
5.5 Registration processes of an Individual from a mobile device 129
5.6 Registration Process Screenshots 130
5.7 Query process from a mobile device 131
5.8 Retrieval Screenshot 131

B.1 Detailed Conceptual Model for MAS-based Digital Ecosystem 149
B.2 Onto2MAS Framework Components 150

List of Tables

2.1 Summary of Agent Concept Models 41

3.1 MAS2DES-Onto Modules 51
3.2 Structural Module Concepts Description 53
3.3 Species Module Concepts Description 55
3.4 Reasoning Module Concepts Description 57
3.5 Interaction Module Concepts Description 58
3.6 System Module Concepts Description 60
3.7 Module Specific Relations 61

4.1 Measures to Generate DESs 90
4.2 Large Size DES Creation in OnToJade Approach 91
4.3 Experimental protocol for Agent Interaction Test 92

Chapter 1

Introduction

1.1 Context

With the rapid advancements of Internet and Web 2.0 technologies, there is a shift in the focus of web applications towards social interaction and collaboration. The Internet and the Web are evolving to a platform for collaboration and sharing of user-created contents. Nowadays, there is a move from the Web as a place of producers and consumers of content to a place of communities where everyone can publish information, interconnect, communicate, collaborate, and share [9]. One aspect in this context is the rising importance of social media. Social media allow the creation and exchange of user generated content that occur at a global level [10]. These media encompass various applications focus on the communication and collaboration among the users. Moving from services provided by a single entity to more complex or integrated multi-stakeholder services requires new approaches for effective consideration of collaboration.

Currently, there are more than 800 active social networking sites and several platforms for people to interact[1]. They are characterized by participation, openness, connectedness, and sense of community [11]. People are living in a rich social media environment, in which they freely and spontaneously generate and share contents of various types as part of their daily activities. Among the shared data on those media, multimedia takes the biggest part. Numerous social media have been created to provide the possibility to share multimedia content. Studies also indicated that 80% of the shared information

[1] http://www.researchandmarkets.com/research/4p6sj4/global_social

on the Internet is with multimedia formats [12]. The global audio and video traffic combined is expected to reach 82% of all Internet traffic by 2018[2]. For instance, more than 500 hours of video are uploaded per minute to YouTube every minute, and 8 billion videos watched per day[3]; and the image hosting site Flicker provides access to over 6 billion of photos[4]. Facebook has over 1.7 billion users who upload 300 million photos per day[5]. These days, video posts are outperforming other types of content across platforms as it is shared nearly four times more than other types of content. Video sharing has grown 140% since 2013[6]. It is even outperforming photo content, once thought to be the best way to engage audiences on social media. In 2015, total engagement of social media users on video content grew 255% over 2014[6]. This shows that sharing of multimedia content is becoming increasingly popular over the Internet. Thus, multimedia data are being captured, stored, and coming from different sources with diversified representations. Yet, the collaboration technology that helps people search, use, and express themselves with these media is lagging behind [12].

There exist collaborative systems such as social media environments and traditional environments (e.g., client/server, peer-to-peer, grid, cloud), that support communication, coordination, and cooperation [13]. Such kind of systems allow groups of users to communicate and cooperate by creating, manipulating, and providing access to a variety of information and resources. Their target is towards making people, information, and resources available to all who need to interact closely with each other. Nevertheless, these collaborative systems present limitations regarding content selection, categorization, aggregation, linking, interoperability, usage control, and privacy.

To overcome these limitations, Digital Ecosystems arise as a new way to handle collaboration in a distributed and heterogeneous environment. The emerge of Digital Ecosystems brings substantial benefits to interested participants allowing collaboration while keeping mutual benefits opposite to traditional collaboration systems which can only provide limited collaboration capabilities [14]. Digital Ecosystems are often described as complex adaptive systems as they are the digital counterparts of biological ecosystems [15]. Multi-Agent Systems (MAS) are regarded as appropriate means for the development and simulation of such complex systems [16]. For the effective and efficient

[2]http://www.tubularinsights.com/2019-internet-video-traffic/
[3]https://www.youtube.com/yt/press/statistics.html
[4]https://www.flickr.com/photos/franckmichel/6855169886
[5]https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/
[6]https://contently.com/strategist/2015/09/14/the-state-of-social-media-content-in-12-charts/

development of any MAS-based systems, comprehensive ontology model is required to represent entities and development framework is needed to deliver the intended system.

As we mentioned above, these days people are more interested in publishing and sharing multimedia contents which drive any proposed collaborative environment should consider this reality. Hence, to overcome these limitations of traditional collaborative environments and meet the increasing interest in sharing multimedia resources, we propose **MMDES**, a **M**ulti**M**edia-oriented **D**igital **E**co**S**ystem. **MMDES** is a special type of Digital Ecosystems that focuses on sharing multimedia resources (data, methods, API/services, processing capacities, and cache and storage capacities) among interested participants.

1.2 Motivating Scenario and Challenges

To motivate our work and elaborate the main requirements addressed by the new collaborative environment, we illustrate the following scenario.

Since March 2014, West Africa has experienced the largest outbreak of Ebola in history, with multiple countries affected (mainly Guinea, Sierra Leone, and Liberia)[7]. In response to the outbreak, the World Health Organization (WHO) activated its emergency operations center to coordinate technical assistance and control activities with other domestic and international partners. To campaigning the control and prevention task, one of the chosen method was using a documentary video. WHO has used a consistent visual style to bring together different chronic disease issues. This will help to create awareness as lack of knowledge amongst the population about Ebola makes it difficult to control this epidemic. Moreover, people are afraid and find it difficult to believe that the disease even exists and the communities there are not familiar with this disease. Thus, the documentary work might help the population to better understand the disease and also to get the support of the rest of the word in tackling the epidemic. In addition, there are a number of local and international organizations engaged in the prevention and control of the virus. These organizations have their own data collection and analysis tools, systems, and strategies. Thus, there has been a call in establishing systems and tools that allow the WHO to respond rapidly and effectively under one unifying strategy.

[7] http://www.who.int/csr/disease/ebola/en/

In this context, we describe two scenarios to illustrate the need for a new collaborative environment

Consider Alice, journalist at the WHO, who has a pressing mission to produce a documentary movie covering the Ebola Virus Outbreak in West Africa. The aim is to discover the patterns of the outbreak in order to better assess the spread of the virus by analyzing up-to-date data on the crisis. Due to high risk of infection and travel restrictions, Alice and her team cannot film the documentary within the infected regions. Nevertheless, she realized that many people and organizations have been publishing and sharing multimedia-based information (i.e., images, audio clips, movie clips, and texts) about the epidemic all over the web (e.g., social networks) using their own means and formats. As a result, Alice decides to collect the information available publicly on different social networks related to the WHO organization, check its integrity, analyze it to provide precautions and predication studies, and aggregate it in order to create her documentary. Before producing her movie, Alice is made aware that the information is: dispersed and available on different social networks, of different nature and formats, unreliable some of the times, unorganized and unordered, and abundant and widespread.

Thus, in order to produce the documentary, Alice and her team need to:

- search, select, and collect relevant data (e.g., movies, audio clips, pictures) published and shared on accessible social networks;
- verify (e.g., check integrity), clean (e.g., eliminate redundancies, eliminate unreliable information), classify and categorize (e.g., by place, time, subject) the data;
- aggregate the classified data to generate contents from multiple sources (e.g., integrate, merge, split);
- extract the knowledge from the aggregated data for analysis purpose (precautions and predictions studies);
- use the discovered knowledge to aggregate the information semantically from different location, services, and users; and
- produce the documentary with minimal effort and expertise in programming and computer science.

To do so, Alice faces several challenges as the existing collaborative platforms cannot solve the above requirements:

- selecting and collecting appropriate content are difficult as most of current platforms (i.e., social medias) support publishing and sharing than collecting;
- platforms provide contents that differ in size and topics which makes automatic classification and categorization complicated;
- there is no common description technique for representing and storing content;
- there is no appropriate means to help entities to establish appropriate collaboration (with respect to a need) according to their resources (data and services);
- existing platforms do not provide appropriate services for efficient discovery and utilization of collective knowledge of users;
- users are not in control of their data and usage, which leads to loss of trust among them, drop of privacy, and hinder them from freely sharing and collaborating resources.

Thus, in order to address the need of WHO and in particular Alice documentary movie production, there is a clear and urgent need to propose a new collaborative environment that takes into account the mentioned challenges. To design and develop such collaborative environment, we need first a model to represent the participating entities. Following this, it is very important to have a framework that support to easily and quickly develop the intended system.

1.3 Contribution of this study

The main contributions of this work can be summarized as follows:

- **Ontology for MAS-Based Digital Ecosystem**. The first contribution is the provision of a generic and comprehensive agent-based ontology, MAS2DES-Onto, that aims at addressing interoperability, communication, behavior expression, role, and rule definitions [17]. It allows to model MAS and Digital Ecosystems independent of any application domain.

- **Framework for Development of MAS-based Digital Ecosystems**. To develop the intended Digital Ecosystems for multimedia sharing and collaboration, we provide a framework called Onto2MAS that considers all the process of MAS-based Digital Ecosystems development. This framework is significant in quick and easy development of MAS-based Digital Ecosystems without requiring high level programming skills [18].

- **Prototype**. OnToJade is an implementation prototype of Onto2MAS Framework which consists of different components to validate the the proposed framework and compare it with existing approach [19].

- **Digital Ecosystem for Multimedia Sharing and Collaboration**. MMDES is delivered as as special type of Digital Ecosystem for managing and collaborating for multimedia contents [17, 20–22]. This new collaborative environment addresses the limitation of social media and other collaborative environments in keeping mutual benefits of all its participants and ensuring equilibrium of the system. The development of this systems takes into account the proposed ontology and framework in this work.

1.4 Publications

The following publications support the material presented in this thesis:

- **MAS2DES-Onto: Ontology for MAS-based Digital Ecosystems**. Solomon Asres Kidanu, Richard Chbeir, and Yudith Cardinale. *Submitted paper to the 32nd ACM Symposium on Applied Computing.*

- **Onto2MAS: An Ontology-Based Framework for Automatic Multi-Agent System Generation**. Solomon Asres Kidanu, Corentin Donzelli, Richard Chbeir, and Yudith Cardinale. International Conference on Signal Image Technology and Internet Based Systems, Accepted Paper, IEEE, 2016.

- **MMDES: Multimedia Digital Ecosystem - New Platform for Collaboration and Sharing**. Solomon Asres Kidanu, Yudith Cardinale, Richard Chbeir, Victor De Ponte, Alejandro Figueroa, Ronier Rodríguez, and Carlos A. Raymundo

Ibanez. International Conference on Computational Science and Engineering, pages 393 - 400, IEEE, 2016.

- **Event Extraction for Collective Knowledge in Multimedia Digital EcoSystem**. Minale A.Abebe, Solomon Asres Kidanu, Fekade Getahun, and Richard Chbeir. In AFRICON, pages 1-5, IEEE, 2015

- **EDiM: Ecosistema Digital Multimedia – Plataforma Novedosa de Colaboración y Compartimiento**. Yudith Cardinale, Alejandro Figueroa, Alvaro Parada, Ronier Rodríguez, Solomon Asres Kidanu, Richard Chbeir. III Conferencia Nacional de Computación, Informática y Sistemas (CONCISA 2015). pages 14 - 25, 2015. ISBN: 978-980-7683-01-2.

- **A Multimedia-oriented Digital Ecosystem: a New Collaborative Environment**. Solomon Asres Kidanu, Yudith Cardinale, Gilbert Tekli, and Richard Chbeir. International Conference on Computer and Information Science, pages 411 - 416, IEEE, 2015.

1.5 Organization of the Thesis

The chapters of this thesis report are organized as follow:

In **Chapter 2**, we provide state of the art about Digital Ecosystems. We start with definitions, perspectives, components, main characterise, and different categories of Digital Ecosystems. We also provide existing frameworks and architectures of Digital Ecosystems. In this chapter, we investigate existing ontology-based MAS methodologies; and more importantly, agent concept modeling in MASs and Digital Ecosystems are studied and analysed.

Chapter 3 provides an ontology model for MAS-based Digital Ecosystems called MAS2DES-Onto. We start with the requirements for modeling MASs and Digital Ecosystems which are initiated from the essential characteristics of both systems. Next, we present MAS2DES-Onto, an ontological model that consists of all the essential aspects of agents in the context of MAS-based Digital Ecosystems. This ontological model has five modules. Each concept of the modules is defined and a relationship between concepts is clearly indicated.

In **Chapter 4**, we start with the need for framework to efficiently and effectively develop complex systems like Digital Ecosystems. We then present in detail Onto2MAS Framework to support the development of MAS-based Digital Ecosystems; with its three main components and OJ language followed by OnToJade prototype, which is an implementation of Onto2MAS. Experimental tests and results are also included for comparing and evaluating the prototype.

Chapter 5 gives **MMDES** (MultiMedia Digital EcoSystems) as a special type of Digital Ecosystem for multimedia sharing and collaboration. First, we describe social media and collaborative platforms with their limitations. Then, we present overview of MMDES with its design and generation processes. We also provide the mechanism to knowledge and query management and ensuring equilibrium in MMDES. Finally, we present deployment of MMDES for a mobile platform.

In **Chapter 6**, we provide summery of our work, and suggests possible future research directions.

Chapter 2

Related Work

2.1 Introduction

With the growing maturity of information and communication technologies (Internet of things, cloud computing, social networking, etc.), systems have been interconnected within growing networks, yielding new services through a combination of the system functionalities. This leads to an increasing complexity for managing, designing, and developing such systems [23]. A key challenge in modern computing is to develop systems that address such complexity in an efficient way [15]. Thus, the concept of "complex system" has been mentioned in different works to describe such systems.

A complex system is a system featuring a large number of dynamically interacting components and systems that collaborate to create a functioning whole [24]. The phenomena of complex systems emerges as a consequence of multiscale interaction among the components and their environments [25]. The key feature of complex systems is that cooperative interaction of the individual components determines the emergent functionalities and behaviors, which individually do not exist [25]. The components, often called agents, typically exhibit self-organization without centralized control mechanism that governs system behavior and interaction in complex systems [26]. Most often, the components are physically and functionally heterogeneous [27]. They are dynamic in which they can evolve, adapt, and transform with a changing environment and in response to external or internal pressures [25, 28]. Thus, determining how various systems are pulled together to accomplish a joint mission is one of the contests facing complex systems. The issue

of integrating constituent components and systems is critical as they are independent systems with multiple objectives [29]. There are several existing approaches aimed at addressing this complexity such as System-of-Systems and Digital Ecosystems [15, 28].

System-of-Systems describe the large scale integration of many independent self-contained systems to satisfy global needs [29]. It is envisioned as a system assembled of other systems so as to offer the capabilities needed to perform roles assigned to the bigger goal of an enterprise [29]. Heterogeneous and distributed systems are involved in System-of-Systems. These multiple systems are likely to exhibit operational and managerial independence with emergent and evolutionary behaviors [30]. These systems do not share a common conceptual basis, are not built for the same purpose, have no common control or management, and evolve at different rates subject to different pressures and needs [31]. Furthermore, the underlying network, communication and information exchanged in System-of-Systems are designed based on traditional distributed system environments like peer-to-peer (P2P), client/server, grid, and cloud. These environments have several constraints to cope with supporting collaboration and cooperation among the various systems [20]. To deal with these issues, new approaches have been identified. One of the new and recent paradigm to address complex systems is Digital Ecosystems.

Digital Ecosystem is a metaphor inspired by natural ecosystems which describes a set of components and systems that are interacting and collaborating for their mutual benefit [15, 32]. It is one of the approaches to develop systems that address complex, dynamic problems, and cope with uncertain environments [33]. This kind of systems has many of the same multiple characteristics as those of naturally occurring in complex systems [25]. Digital Ecosystems would be more effective than traditionally inspired approaches to represent and simulate complex systems, because it is a digital counterpart of biological ecosystem and would be built upon the scalable, evolving, and self-organising properties of biological ecosystems [34].

Regarding the development and simulation of complex system (i.e., System-of-Systems and Digital Ecosystems), Multi-Agent Systems (MASs) appear as an appropriate model [16]. A MAS is composed of autonomous agents that interact each other in a given environment to achieve their goals [35]. An agent-based perspective is important for complex

systems in defining what is potentially a common conceptual framework for system architectures. This is mainly due to the characters that agents have. Agents are able to perceive their environment, recognize and understand changes in that environment, respond to them in a timely manner [35]. The other important characteristic is the agents' ability to form distributed systems and their ability of autonomous problem-solving and self-organized decision making [36]. Moreover, agents and their interactions can be easily simulated using different tools [37]. These all assure that agent and multi-agent technologies have promised to address complex systems [3].

However, a key problem is the lack of a formal methodology and approach for the developing and managing such complex systems [30, 38]. It is known that development of complex systems claims for special programming skills and it is a hard and time consuming task. In order to design and deliver effective complex systems, a framework is required for guiding and coordinating the development tasks [39]. A framework is a common, stable, and coherent structure, which can be concertized or implemented in different ways.

In this chapter, we first present definitions, components, characteristics, and categories of Digital Ecosystems. Following this, we cover related works about frameworks and architectures of Digital Ecosystems. Then, we provide a review of agent concept representations in MASs and Digital Ecosystems and the chapter ends up with a conclusion.

2.2 Concepts of Digital Ecosystems

2.2.1 Overview

Nowadays people are recognizing that they are living in a digital environment which can be built as being analogues to the ecological ecosystem. This is due to the rising of the Web technologies and its significant impact on people, organizations, and businesses [40]. As a result, it is impossible to deny the enormous impact of the web on economy, social, and political affairs of the globe [40, 41]. By understanding this fact, the European Commission initiated a research vision by 2000 in relation to the digital environment, known as Digital (Business) Ecosystem. The aim was developing a knowledge-based economy for sustainable economic growth of Small and Medium sized

Enterprises (SMEs) with support of Information and Communication Technologies [42]. Digital Ecosystems have emerged with the purpose of enhancing communication among SMEs within the world of Business Ecosystem [43]. The goal of such Digital Ecosystem was to improve the efficiency of the communication among enterprises and to structuralise the existing business ecosystems. These ecosystems are based on a technological infrastructure to mediate the formalization of knowledge in SME networks, the creation of software services, and the business-to-business interactions among the SMEs. Then after, the term *Digital Ecosystem* has been viewed in various ways in the literature.

Since Digital Ecosystems have been conceived as a business supply chain ecosystems, they have been considered as platforms in which systems of each company interact with those of other companies in the supply chain [44]. In this context, a Digital Ecosystem is defined as a digital infrastructure aimed at creating a digital environment for those networked organizations. While in other contexts, it is considered as a service for enabling customers to use existing e-business solutions and is widely linked with to support business ecosystems [45, 46]. However, these considerations have in common, the existing and most frequent reference about digital ecosystems: investigate and exploit properties of biological and other complex systems [41]. Thus, Digital Ecosystems are the digital counterparts of biological ecosystems, which are considered to solve complex, dynamic problems by offering an approach for conceptualizing, designing, and organizing suitable environments [43]. Digital Ecosystems, as an approach, integrate and use the concepts of a given natural domain to the digital world, reproducing or interpreting some of the mechanisms of natural ecosystems in terms of interaction, equilibrium, mutual benefit, etc. Because of this ground, the concept of Digital Ecosystems is viewed as a new way of perceiving the increasingly complex and interdependent systems being created today [44]. With these general conceptions of Digital Ecosystems, we found numerous works in the literature that point out definitions of Digital Ecosystems from various viewpoint.

Chang and West [47] define a Digital Ecosystem as an environment which has a number of characteristics as follow: *"an open, loosely coupled, domain clustered, demand-driven, self-organizing, and agent-based environment in which each species is proactive and responsive for its own benefit and profit"*. Boley and Chang [43] address it from species aspect in their work: *"a digital ecosystem is a digital environment populated by digital species or digital components which can be software components, applications, services,*

knowledge, business processes and models, etc.". In [48], the concept is described as: *"... complex of digital communities consisting of interconnected, interrelated, and interdependent digital species situated in a digital environment that interact as a functional unit and are linked together through actions, information, and transaction flows"*. Digital Ecosystem is also mentioned as a collaborative environment in [41], *"a self-organizing digital infrastructure aimed at creating a digital environment that supports the cooperation, the knowledge sharing, the development of open and adaptive technologies and evolutionary business models"*. Liang and et al. [49] define it as *"self-organizing systems which can form different architectural models through swarm intelligence, where local interactions between agents determine the global behavior"*. This definition clearly stated that Digital Ecosystems are systems of interdependent systems. Sulonen [50] also explains a Digital Ecosystem as a kind of community where different parties with different levels of interests and resources participate and provide resources in a give-and-take way that creates benefits for all.

From the above definitions and explanations of Digital Ecosystems, we have mainly observed that: (i) most of them focused in defining a Digital Ecosystem as a complex system derived from an environment and species existing in this environment, and (ii) there are different perspectives of understanding the Digital Ecosystem concept: ecological, technological, and complex systems perspectives. The perspectives impact the type of designed and developed Digital Ecosystems. In this thesis work, we mainly account the Digital Ecosystem as one of the approaches to address the design and development of complex systems by mimicking some of the characteristics of biological ecosystems. Apart the perspectives, we can see that a Digital Ecosystem has two main constituents (i.e., species and environment) with a number of characteristics. On the following subsections, we elaborate this concept.

2.2.2 Components of Digital Ecosystems

Biological ecosystems are mainly composed by "species" and "environment" [51]. Species need to interact with each other and keep mutual benefit in the environment where they exist. The environment supports the needs of its species so they can continue generation after generation. Likewise, a Digital Ecosystem is composed of a digital species and a digital environment. In the following paragraphs, we address these core components.

2.2.2.1 Species

The digital species are analogous to the biological species, living organisms that are autonomous, viable, and self-organizing [52]. They are proactive and adaptive entities that populate the Digital Ecosystem environment [53]. They come to an ecosystem of their own demand and are heterogeneous; in fact their diversity is what makes the Digital Ecosystem viable [54]. They are motivated by their own benefit and carry out tasks that relate to their own profit and existence of the ecosystem [32]. These species can play different roles simultaneously in the Digital Ecosystem being provider, consumer, executor, coordinator [47]. They interact with other species through shared commonly agreed conceptual models and following rules of the Digital Ecosystem [43]. The rules are defined to determine how species have to participate, share, and access resources in the environment. Species can also form communities according to their preferences and domain of interests and also participate in communities of their own initiative.

In Digital Ecosystems, digital species interact through information flows. The information flow is what keeps the digital species alive and also dictates their evolution. Through this, species could be aware about the rest of the species and the environment. The perception helps species to know about available and applicable services. This makes species adapt itself to changes in the Digital Ecosystems. Moreover, the perception of species determines the way they collaborate with others and how to take necessary actions in the ecosystem. Additionally, species provide an ecosystem with dynamism, efficiency, and stability. These species can be represented in a form of software agents in the Digital Ecosystem. Agents are functionally parallel to the organisms of biological ecosystems, including the behavior of migration and the ability to be evolved [15].

2.2.2.2 Environment

Digital environments are environments in which digital species exist, interact, and evolve [55]. According to Hadzic and Chang [52], a digital environment is an environment in which digital species jointly live, function, and relate. Boley and Chang [43] also discuss an environment which contains human individuals, information services as well as network interaction and knowledge sharing tools along with resources. Briscoe et al. [33] consider an environment as population's habitat where different species live

together and interact. This environment provides the necessary infrastructure and underlying technologies and services that supports the interacting elements in it [56]. These technologies provide services that are required for Digital Ecosystems. The environmental setup includes the mechanisms for the composition, the evolution, and the migration of the digital species among the different habitats [41]. The required environment for the Digital Ecosystem should have a number of characteristics in a way that enables the species to live, interact, benefit, and survive likewise the biological counterparts. In the following, these characteristics are discussed.

2.2.3 Characteristics of Digital Ecosystems

The characteristics of Digital Ecosystems are mainly derived from the essential components of ecosystems (i.e., species Seland environment). Based on these components, various researchers in the area discuss different characteristics of Digital Ecosystems which are essential to evaluate, design, and develop them [20, 43, 47, 54]. In the following, we briefly describe the main ones.

Self-organization. Ecosystems have been described as self-organizing systems [51]. This characteristic describes participants of a Digital Ecosystem as species capable of acting autonomously, making decisions, and carrying out tasks in the ecosystem. They are coming to this environment to share and collaborate with others for mutual benefit. The focus is very much on the autonomy of individual species and hence the global properties of the ecosystem emerge through self-organisation.

Interaction and engagement. Digital Ecosystems can be described as a network of species that interact with each other [57]. It is an interactive system established between a set of active agents and an environment within which agents become involved in their common goals [58]. This characteristic presumes that there are several distinct species, i.e., a Digital Ecosystem is formed by a group of species. These species of a Digital Ecosystem should need to interact and engage within each other to find interesting things and to share resources.

Balance. Digital Ecosystems should have this characteristic to keep the benefits of all participants in a mutual way and act responsibly for the safety and sustainability of its environment. Retain balance is important for the continued survival and existence of

species as well as the stability of the environment [56]. To ensure balance, different rules can be enforced in the ecosystem. For instance, setting a minimal threshold value for the amount of contributions in term of resources in a given time frame. These measures would help punishing free riders or selfish species in the ecosystem.

Domain-clustered. This characteristic states that species have something in common or share the same interests [47]. Having this, they join the Digital Ecosystem environment where they have common interests, without external pressure. Species contribute to the Digital Ecosystem community of their desire. They also choose their area of interest and the extent to which they would like to contribute. Thus, species cluster together with similar domain of interests, but are free at any time to change their associations.

Loosely-coupled. As an attribute of a Digital Ecosystem, loosely-coupled enables to reduce the inter-dependencies across species and supports open relationship among species [43]. This would reduce the probability of changes in one species affect others in the ecosystem. In addition, this characteristic is essential to increase flexibility in adding new species, replacing species, and changing operations within individual species.

Demand-driven. This characteristic indicates that species are coming to the ecosystem fascinated with the benefit/profit they will gain from it [59]. Species are not told or forced to participate instead they join the ecosystem from a perceived mutual interest of the parties collaborating. Hence, species join in a community based on their own interests and determine their own requirements.

Dynamism. It is one of the key properties in complex systems like Digital Ecosystems to describe their accommodation of changes in terms of various factors [60]. Species are not in a static condition as their attributes, perceptions, interests, relations, and interactions with others might evolve through time.

Scalability. Digital Ecosystems should consider scalability and robustness among characteristics to exploit the properties of the biological ecosystem [61]. Scalability refers to the ability of the Digital Ecosystem to be enlarged to accommodate growth.

Open. Ecosystems are open systems as they do not have a clear boundary [51]. Accordingly, a Digital Ecosystem is assumed as a free transparent environment, where everyone is invited to join except dangerous species that have the intention of causing damage to the community or not willing to keep the rules of the ecosystem.

2.2.4 Categories of Digital Ecosystems

According to various definitions and views of Digital Ecosystems, several works have been proposed and studied different classifications of them [60, 62]. In the following, we discuss different categories of Digital Ecosystems from three perspectives, namely business, knowledge management, and services.

2.2.4.1 Digital Business Ecosystem

The Digital Business Ecosystem (DBE) results from the combination of Digital Ecosystem and the business world. Applying the concept of Digital Ecosystem in the economic field, it comes up the DBE paradigm, i.e., an environment in which businesses can interact with each other in effective and efficient ways. According to [46], being part of the DBE means that a company is aware of the range of products and services available from all of the other partners and can easily match them with its business requirements. At the same time, its products and services are also being showcased to other companies so they can identify it as a potential business partner. The objective is to enable SMEs to create, integrate, and provide services more efficiently and more effectively [54].

The DBE consists of an infrastructure based on a P2P distributed software technology, to transport, find, and connect services and information over Internet links and enable networked transactions and distributions of all the digital entities present within the infrastructure [46]. The DBE is designed based on the following principles: (i) no single point of failure or control, DBE should not be dependent upon any single instance or actor; (ii) equal opportunity of access for all; and (iii) scalability and robustness [63]. These principles imply a fully decentralised architecture; the design of a P2P structure that is robust, scalable, self-organising, and self-balancing and that embeds scale-free networks and mesh topology dynamics. In this ecosystem design, there is no central repository or database and there is no node/actor that has a privileged or full view of the ecosystem. However, the evolutionary architecture and distributed intelligence enable the "migration" of the formalised knowledge and the software services where there is a greater probability of their use. The DBE is a fully distributed information structure which is essential for keeping the flexibility of the system and for supporting the dynamic connections and re-organisation among the social, technical, and knowledge networks.

The environment in a DBE is the economy, with the actors of the system considered as agents [64]. A population is a group of agents subject to evolutionary change within their respective communities. As they consider the actors of the system as agents, we can consider the DBE as a MAS.

2.2.4.2 Digital Knowledge Ecosystem

Digital Knowledge Ecosystem (DKE) is another category of Digital Ecosystems with the objective of making easier knowledge flow among the entities within the ecosystem [63]. Wikipedia could be considered as a typical example of DKE as it has many of the necessary properties [44].

Wikis are systems allowing large-scale collaborative knowledge creation and distillation [65]. The experience of the Wikipedia project, the most visible and well-known wiki system, has demonstrated that large numbers of contributors can jointly create, review, revise, edit, and manage large amounts of high-quality digital content. Wikipedia has become a proven and accepted collaboration tool used by increasing numbers of organizations to support their collaborative knowledge creation processes. Wikipedia, as a platform of supporting joint knowledge creation, can be regarded as a good example for Digital Knowledge Ecosystem. It comprises large groups of users who cluster together around common areas of activity, as well as interact with members of other clusters in ways characteristic of Digital Ecosystems. Clusters of authors in Wikipedia could be considered as species of the ecosystem. Authors who share the same area of expertise, regardless of their role in the content authoring process, are considered to belong to the same species. It is conceivable to distinguish the actors in this system by the role they take on within the Wikipedia, such as writers, editors, or tinkerers. Being volunteers, users choose to contribute to the Wikipedia of their own volition, they choose the categories and articles to which to contribute, and they choose the extent and nature of their contributions. They cluster together with others of similar expertise and interest, but are free at any time to change their association. Wikipedia contributors pro-actively and independently take the actions they consider necessary, and self-organise to achieve goals as and when the need arises. Thus, we state large volunteer-contributed Wikipedia as a Digital Ecosystem. The Digital Ecosystem for Agriculture and Rural Livelihood

(DEAL) [66] can also be considered to be a DKE, where the knowledge sharing and management is for the benefit of the society of rural agriculture in India.

Hence, the aim of DKE is to foster the dynamic evolution of knowledge interactions among entities to improve decision-making and innovation through collaboration [15]. It is considered as a kind of Digital Ecosystem towards enabling self-organization and dynamic evolution of knowledge interaction among entities in response to changing environments.

2.2.4.3 Digital Service Ecosystem

The Digital Service Ecosystem (DSE) is the third type as a value propagating ecosystem of people, technology, other internal and external service systems, and shared information [67]. The aim is to establish an environment where set of organizations can provide and consume services in a coordinated way [68]. This kind of ecosystem is to incorporate and construct a capability infrastructure that synergizes and improves organizations collective intelligence to adapt to new business visions and opportunities [69]. Within a DSE, organizations usually adopt one or more of the following roles [70]: service provider, service consumer, or service mediator. DSE is formed by applying the generic concepts of Digital Ecosystem on designing and developing service ecosystems [71]. Some attempts under this category are Semantic Service Retrieval Platform for the Digital Ecosystems Environment [72], Service Descriptions in Digital Ecosystems [73], and Business Modelling for Service Ecosystems [74].

2.2.4.4 Discussion

We reviewed the existing Digital Ecosystems to understand the extent to which these diverse systems resemble to biological ecosystems. These systems vary in the way addresses the ecosystem concept, and frequently the word "ecosystem" is merely used for branding purposes without any inherent of ecological properties. We consider the Digital Ecosystems that exploit the properties of biological ecosystems (like interaction, self-organization, balance). However, we noticed that the main characteristics of Digital Ecosystems have not been fully explored in existing Digital Ecosystems. Moreover,

there is no holistic understanding about the Digital Ecosystems. In the following, we will present proposed frameworks and architectures for the Digital Ecosystems.

2.3 Frameworks and Architectures of Digital Ecosystems

Since Digital Ecosystems are complex and involve large number of agents, there is a need for system modelling techniques and methodologies to guide the process of ecosystem design. Without adequate techniques to support the design process, such systems will not be sufficiently reliable, maintainable, or extensible, and will be difficult to comprehend. The modelling techniques are required to describe well the external and internal perspective of Digital Ecosystems with the various entities participating in the ecosystem. In attempting to propose an approach and framework that provide adequate support for the process of designing and generating Digital Ecosystems, we explored existing works. In the following, some of these woks are illustrated to better understand the Digital Ecosystem concepts beyond the theoretical outlook.

2.3.1 Dynamic Agent-based Ecosystem Model (DAEM)

César A. Marín et al. [75] propose a Dynamic Agent-based Ecosystem Model (DAEM) which combines ideas from natural ecosystems and MASs for business interactions and for searching the best services in the environment. This work provides a framework to leverage the strategic concept of a Digital Business Ecosystem (DBE). Authors detail that interactions are fundamental to the creation of ecosystems and describe the development of adaptive behaviours through agent technology. In DAEM, an agent represents an organisation which acts as supplier of one service and consumer of another. The whole collection of agents offering and requiring different services forms the multi-agent business ecosystem. DAEM also defines an environment under which agents exist and evolve. In the context of this work, the environment is a virtually observable surface where inhabitants wander across and encounter others in order to interact. However, the authors focus on the theoretical aspects of DAEM. It is just a synthesis of ideas from biological ecosystems and MASs. There is no practical implementation of their ideas for testing and validation.

2.3.2 Framework for Business Ecosystem Analysis and Modeling (BEAM)

Chunhua T. et al. [76] present a comprehensive framework which has the following layers: service ecosystem modeler, service ecosystem dynamics simulation, and service analysis. The framework integrates different methods as follows: MAS as the computation framework for business ecosystem modeling and analysis capabilities, game theory as the entity behavior model, value network as the systematic modeling method, and role-based paradigm for characterizing ecosystem entities to allow the evolution of the ecosystem. Though this work deliverers a detail architecture and meta-model, it lacks detailing the way agents are represented and the means to make agents are real representatives of an entity in the context of Digital Ecosystems. Moreover, this framework does not explain the model and mechanism of ecosystem evolution.

2.3.3 Digital Business Ecosystem (DBE) Framework

The DBE Framework is for a business ecosystem developed in European Commission integrating SMEs [42]. This framework mainly comprises two layers: business and digital. The Business Ecosystem layer is upon the hardware and software architecture, since they are real work enterprises. This layer consists of interacting organizations and individuals of the economic community (i.e., organisms of the business world). The second and more important aspect of this model is the Digital Ecosystem Layer, which is composed of three sub-layers: coordination, resource, and service layers. The Digital Ecosystem layer shows the relationships between the SMEs and supplies technical support to facilitate their transactions. The framework also shows the necessary technical infrastructure, legal framework, and political support required for the development and deployment of DBE.

This DBE structure is to represent business-to-business interactivity supported by a software platform, which should have the desirable properties of a natural ecosystem and follow a Service-Oriented Architecture (SOA) approach that adheres to the original SOA principles. With this infrastructure, the ecosystem can achieve evolution, self-organisation, and a self-optimising environment built upon an underlying SOA. However, the SOA architecture has limitations to meet the requirements of Digital Ecosystems as mentioned in [77].

2.3.4 Service-Oriented Peer-to-Peer Architecture for Digital Ecosystems

SOA paradigm is becoming popular for modelling and building distributed systems in heterogeneous, decentralised, and open environments. This paradigm enables interoperability between different software applications running on a variety of platforms and frameworks. However, existing and proposed SOAs are usually based on centralised components, such as service registries or service brokers. Authors in [78] propose a decentralised SOA built on top of a self-organising P2P infrastructure. The P2P infrastructure allows an efficient implementation of the SOA operations, i.e., service registration and deregistration by the service provider and service discovery and consumption by the service consumer. This decentralization is crowned to SOA because of its union with P2P infrastructure. Thus, the authors claim that this new form of SOA made it capable to support Digital Ecosystem where the environment is decentralised, open, and heterogeneous.

In spite of the new feature of SOA thanks to P2P, the work has no further complement about the type and model of the Digital Ecosystem for their proposition. They just imitate the term Digital Ecosystem, detail their design proposition of a decentralized SOA architecture, and then conclude that this architecture fits the characteristics of Digital Ecosystems.

2.3.5 Ecosystem-Oriented Architecture

By arguing SOA is not good enough to support a Digital Ecosystem, two works proposed an ecosystem-oriented architecture by extending SOA. These two works are presented shortly as follows.

Ferronato P. [77] proposes a new architectural style, called the Ecosystem Oriented Architecture (EOA). The motivation for this work is SOA does not support the basic characteristics of a Digital Ecosystem environment [77]. Then, the author extends SOA with capability of supporting dynamic evolution by adopting P2P architecture as the network infrastructure. EOA provides the description of digital components and processes that are involved in the ecosystem. In EOA, all components interact together,

crossing organizations' boundaries and forming a Digital Ecosystem that connects different systems and exchange information using common data representation formats. All EOA services are deployed on a distributed P2P platform. The decentralized P2P architecture defines a topology and a replication schema that depend on a set of collaborative peer nodes. The required architecture needs a mechanism to allow participants to publish whatever model and to investigate which is the most adequate for their needs. However, the proposed architecture tries to address only the issue of centralized control in SOA but not other essential issues of Digital Ecosystems in its architecture.

Gerard B. and Philippe W. [61, 79] also discuss EOA by extending SOA with Distributed Evolution Computation (DEC), which allows services to recombine and evolve over time constantly seeking to improve their effectiveness for the user base. Authors define the architectural principles of Digital Ecosystems by combining elements from mobile agents systems, distributed evolutionary computing, and SOA. In this architecture, each service is a habitat represented by an agent. The automatic combination of agents creates a Digital Ecosystem. The agents interact, evolve, and adapt over time to the environment, thereby serving the ever-changing requirements of the user base. These agents recombine and evolve over time, constantly seeking to improve their effectiveness. Additionally, the migration of agents is optimized by genetic algorithms to find the place where they are useful in fulfilling requirements. The ability to migrate is provided by using the paradigm of agent mobility from mobile agent systems.

This work also provides EOA for the Digital Ecosystem but in a different way by introducing a mechanism including the behaviour of migration and the ability to be evolved. To a certain extent, the EOA simplifies the design of Digital Ecosystems and increases the efficiency of systems' functioning. Still the base of this work is SOA (with a support of other technologies) which assumes everything in a form of services. In addition, this architecture targets to address the DBE like most of the works. More importantly, this work only focuses on assuring the self-organization characteristic of agents in the DBE environments.

2.3.6 Modeling Framework for Service Ecosystems

The work presented in [74] proposes a modeling framework for service ecosystems consisting of three layers: Business Layer, Agent Society Layer, and Physical Network Layer.

In fact, this work focuses on the Business Layer and proposes a Business Model for service ecosystems. Authors present the Business Model from three levels, i.e., from service world to service ecosystem and finally to service individual. Service world consists of a set of ecosystem services and their relationships. Service ecosystem has a set of service communities and their habitats. A service individual represents a service with its capability, requirements, reaction strategy, and knowledge memory. An individual could make a decision to move to other groups of species within or outside its community for more benefits. This depends on how an individual evolves in a new environment and meets requirements of a community as each community has its own rules under which each individual decides its own behavior. Thus, each individual is supposed to have a strategy to adapt its behavior to the environment. However, this work is not complete as it only focused on the first layer of the model. It lacks mapping the business model to the agent society layer to establish an execution platform for service ecosystems and validate its effectiveness.

2.3.7 Discussion

Digital Ecosystems have been designed as systems exploring properties of natural ecosystems in different domains. Various frameworks and architectures have been proposed to model Digital Ecosystems. They are mainly modeled with two approaches, namely SOA extending approaches and MAS based approaches. These approaches are addressed either from natural ecosystem imitating perspective or agent society perspective. MAS technology is exploited pervasively to model self-organizing behavior of business ecosystems [75], which combines ideas from biological ecosystems and MASs. MASs also are used for proposing a comprehensive framework to integrate business ecosystem modeling and analysis capabilities [76]. In EOA, the services are modelled as software agents and the Digital Ecosystem is made of MASs with distributed evolutionary computing to combine suitable agents available in the ecosytsem [80].

The digital environment of business ecosystem is imagined as an open agent system in which the components are intelligent, autonomous, and heterogeneous [81]. These characteristics make agents better equipped to handle different kinds of dynamics that can result from their interaction within changing environments. Thus, most of the frameworks and architectures of Digital Ecosystems specify MAS as a technology for the

development and simulation of the Digital Ecosystems [16]. However, the existing architectures are focused on addressing business ecosystem. This leads to the development of a number models for business ecosystem [75]. Besides, most of the authors come up with a theoretical framework for Digital Ecosystems without going further for checking its practical applicability. Furthermore, MAS development requires the addition and removal of agent instances quickly and a means for structuring and restructuring the organisation of MAS "on the fly". Existing modeling languages and ontology-based MAS methodologies are presented in next sections to investigate whether they support this requirement or not.

2.4 Ontology-Based MAS Development Methodologies

2.4.1 Introduction

MASs have become one of the most promising technology exploited in several application domains [82]. In order to help the knowledge representation in MASs, a battery of modeling languages and methodologies have been proposed in the literature [83]. Methodologies offer a standard way to analyze, design, and construct MASs [84]. MASs are appropriated as a means for modeling complex systems such as Digital Ecosystems. Ontologies enable the agents to fully understand the knowledge domain and to identify possible links between different ontology concepts [7]. However, in the context of Digital Ecosystems, there is still a need for general conceptual models and methodologies to represent the specific characteristics of Digital Ecosystems. This ontology can be used for MAS-based Digital Ecosystems modeling and development, considering the simplicity, efficiency, and speed in code generation, that it offers to develop MASs and Digital Ecosystems [85]. However, many existing MAS methodologies explore ontologies to reach the MAS goals and to formally represent its knowledge [86, 87]. Most of them require the usage of ontologies to support the MAS and to define its domain knowledge.

In this section, we present some of these MAS development methodologies that integrate ontologies for knowledge representation and sharing to support application domains such as MAS-CommonKADS [88, 89], MESSAGE [5], INGENIAS [6], MaSE [90] PASSI [91], and MOMA [92].

2.4.2 MAS-CommonKADS Methodology

MAS-CommonKADS [88, 89] is an extention of a methodology for knowledge engineering called CommonKADS[1] to address the distributed nature of MASs. The purpose of the methodology is to address for the agent model with social, cooperative and cognitive aspects, and adding a cooperative model and a system model to consider the organizational aspects of the MAS. MAS-CommonKADS uses the object-oriented notation to structure systems, use cases to capture requirements, and standard protocol specification techniques and message sequence charts to describe agent interactions. MAS-CommonKADS considers ontologies to represent the knowledge of the application domain and an agent local domain-related knowledge. It illustrates the use of ontologies for knowledge representation during agent modelling. However, it does not recognise the role of ontologies in the design and development of agents.

2.4.3 MESSAGE Methodology

The MESSAGE methodology [5] covers the MAS analysis and design phases of the software engineering cycles to develop complex distributed applications. The analysis and design process adopts the Rational Unified Process and extends UML to support the modelling of concepts. Analysis focuses on describing the organizations involved in the MAS, agent goals, and the roles defined to satisfy them. MESSAGE analyses results in a collection of models describing the system to be developed and its environment. Five views have been defined from the models to help the designer focus on the coherent subsets of the MAS (details about MESSAGE views and concepts are available in Section 2.5.2.6). During the design, the analysis models are refined into computational entities that can be implemented in an agent platform. Agents are identified during design based on the description of the organizations and are assigned to all the organization's roles in the multi-agent application. Two distinct phases of design are considered in this methodology: high-level and detailed design. The first phase involves identifying an agent architecture, assigning roles to agents, and describing how the application services are provided in terms of tasks. During the second phase, the refinement process continues to determine how entities can be implemented. The design description is an implementation-independent conceptual description of the system. However, some

[1] https://commonkads.org/

of the more difficult issues related to the dynamic behaviour of MASs are not explicitly addressed.

Similar to MAS-CommonKADS, MESSAGE uses ontologies as the representation mechanism for modelling an application domain knowledge and an agent local domain knowledge. MESSAGE makes it possible for agent reasoning to use ontology-based knowledge at run-time. It does not include an apparatus for ontology-sharing between the agents. More importantly, the methodology does not recommend the generation of agents from ontological concepts.

2.4.4 INGENIAS Methodology

INGENIAS is both a methodology and a set of tools for the development of MASs [6]. As a methodology, it provides a notation to guide the development process of a MAS from analysis to implementation. It is based on five metamodels that define the different views and concepts of a MAS (more about views in Section 2.5.2.7). INGENIAS is supported by a set of tools for modeling (graphical editor), documentation, and code generation (for different agent platforms). The development process of INGENIAS follows the principles of Model-Driven Development (MDD). It bases the development on the specification of the models of the system and the automated generation of other artifacts, like documentation or code, from these models. These activities are assisted by agent-oriented software tool called INGENIAS Development Kit (IDK)[2]. This tool allows to edit consistent models and to generate documented code in different languages such as JADE[3]. Despite these aids, the application of MDD principles is not completely considered in this methodology since there is nothing mentioned about Model Transformations (MTs) stage of MDD. A MT transforms a source model, following a source meta-model, into a target model, following a target meta-model. It is obvious that exhibiting transformation of meta-model into a program would help clarifying the MAS development process itself. In fact, this work is not considered organizational dynamics (i.e., how agents can join or leave the system, how they can form groups dynamically, what their life-cycle is). As INGENIAS is an extension of MESSAGE, it shares the same constraint in considering ontology through its MAS development process.

[2] http://ingenias.sourceforge.net/

[3] Java Agent DEvelopment Framework (JADE) is a software framework for the development of intelligent agent, implemented in Java [93]

2.4.5 MaSE Methodology

The Multi-agent Systems Engineering (MaSE) methodology is a general purpose methodology for developing heterogeneous MASs [90]. It is a start-to-end methodology that covers from the analysis to the implementation of MASs. MaSE uses a number of graphically based models to describe system goals, behaviours, agent types, and communication interfaces. MaSE also provides a way to specify architecture-independent detailed definition of the internal agent design. MaSE uses conventional Object-Oriented modeling techniques. The main goal of MaSE is to guide a designer through the software lifecycle from a documented specification to an implemented agent system, with no dependency of a particular MAS architecture, agent architecture, programming language, or message-passing system.

MaSE is an improvement over MESSAGE and MAS-CommonKADS in that it recognises the essential role of ontologies in agent communication. In particular, it requires the developer to formulate the exchanged messages in respect of the concepts obtained from an ontology. MaSE also uses ontologies to support interoperability. It considers the case of agents committing to heterogeneous ontologies, e.g., agents wrap around heterogeneous information sources and it highlights the need for ontological mappings between these local ontologies. However, MaSE did not mention the importance of ontology to develop the required system. The importance of ontology is limited to address interoperability issue.

2.4.6 PASSI Methodology

Process for Agent Societies Specification and Implementation (PASSI) is a step-by-step methodology for designing and developing MASs [91]. PASSI integrates design models and concepts from Object-Oriented software engineering and artificial intelligence approaches using UML notation. This methodology is made up of five models: System Requirement, Agent Society, Agent Implementation, Code, and Deployment. PASSI also uses ontologies to model both domain knowledge and agent local knowledge. It supports the use of ontologies for agent communication as well. For each agent conversation, an agent developer identifies the ontology that needs to be shared by the communicating agents and defines messages in terms of shared ontological concepts. PASSI does not

provide support for the use of ontology-based knowledge for agent reasoning. Like other methodologies, PASSI does not consider ontology for rapid and simple designing and generating of MASs.

2.4.7 MOMA Methodology

Metadata-based Ontology MAtching (MOMA) is a methodology driven by ontology development [92]. It consists of two main development phases: Ontology Development and Agent Development. During the first phase, domain knowledge is modelled in the form of ontology so that it does not have to be defined in lower level code during agent development. This phase is broken down into three main steps: concept identification, ontology modeling, and code generation. The resulting ontology is then used as a basis for the second phase, which involves the implementation of the agents and the application environment for a specific application. The purpose of the Agent development phase is to implement the ontology and code generated from the ontology into the agent application. The ontology should contain the domain knowledge. It is up to the agent developer to identify, design, and code the agent societies. This work tries to better consider ontology in the development of agents than others. Authors partially achieve the goal to reduce the development effort of MASs by separating the domain ontology from agent development. However, the development process is made by creating large-size Java files and running on JADE platform which takes too much time and requires high level programming skills.

2.4.8 Discussion

Many agent-oriented methodologies have been proposed based on a variety of concepts, notations, techniques and methodological guidelines. Some of these methodologies are derived knowledge engineering, others extend software engineering and agent-oriented methodologies [89].

There are a large variety of works proposing methodologies and approaches to integrate ontologies to MASs for knowledge representation and sharing under specific domain. Nevertheless, none of those works incorporates ontologies to support the development of MASs, independently of specific domains. Moreover, and to the best of our knowledge,

there is no any previous work that proposed an ontology in the global modeling of MASs to develop and instantiate a system in an easy and quick way. This gap in the relationship between ontologies and MASs calls to investigate how an ontology can be used for MAS-based Digital Ecosystems modeling and development, considering the simplicity, efficiency, and speed in code generation. One of the objectives of this thesis work is to provide developers a framework for rapid means to build MAS, by using ontologies. As it is known the development of MASs requires much low-level programming and software development skills in order to develop useful applications.

To better define a conceptual model to support the development process of MAS-based Digital Ecosystem, it is important reviewing existing agent modeling languages from MAS and Digital Ecosystems perspective. Thus, the next section dedicates to present agent concept representation from MAS and Digital Ecosystems works.

2.5 Agent Concept Representations

2.5.1 Introduction

Ontologies can be beneficial to represent the structure of the Digital Ecosystem itself. Agent and multi-agent technologies have promised to address the characteristics mentioned in the Digital Ecosystem arena [3]. This is due to the agents' ability to form distributed systems and their ability of autonomous problem-solving, making decisions, and fulfilling responsibilities [94].

There are key issues that individual agents require to operate in the MAS-based Digital Ecosystem environment. Conceptually, there is a strong parallel between the software agents of a MAS and species of a biological ecosystem [41]. Thus, the agents within a Digital Ecosystem need to be like biological individuals in the sense that they vary, interact, and move [33]. These agents can also change over time, and can be heterogeneous and exhibit different behaviors [81]. Each of these properties contributes to the dynamics of the ecosystem. However, the way in which these individual properties are represented may vary substantially depending on the intended purpose of the system. They should be able to perceive their environment as input which they filter and accept for further processing and to receive messages from other agents. Accordingly, these agents update

their beliefs/knowledge based on both the percepts as well as the information received in exchanged messages, their temporary knowledge about the environment, and other aspects. Then, agents react based on a specific set of rules that describe individual behaviours and engage in a deliberation process, which allows them to adjust their goals and plans, and decide what is the next action to be performed. These agents act and the effects of their actions appear in the environment.

We have identified key issues that individual agents require to operate in MAS-based Digital Ecosystem environments. Therefore, in order to create a model of an agent, one would require:

1. **Structure**. Agents are coming to Digital Ecosystem environments with their own interests. Thus, they should provide information about their properties, characteristics, resources, and the likes; these details are important to well interact with other agents.

2. **Behavior**. This is about representation of the internal state of an agent; agents in Digital Ecosystems are supposed to be proactive and reactive. Accordingly, these agents must update their beliefs/knowledge based on their precepts, information received, knowledge about the environment, and others aspects. Thus, important data structures with a set of their corresponding operations in terms of beliefs, goals, plans, messages, percepts should be represented.

3. **Roles and Rules**. Agents may come to Digital Ecosystem environments to be provider, consumer, or both of resources and services. Hence, the role of agents must be clearly defined to effectively facilitate their sharing and collaboration in the system. Agents may define different types of rules to play appropriately their roles and balance mutual benefits. Following this, agents react based on a specific set of rules (that dictates their behaviors) and engage in a deliberation process. Therefore, means of encoding rules and roles that express reactive behaviour is needed.

4. **Reasoning**. Agents in Digital Ecosystems will need to be able learn and reason to keep up with a dynamically changing environment. Means of representing the functionality that corresponds to their proactive behaviour is important for autonomous agents. This makes agents capable to adjust their goals, plans, and

actions to be performed; and also to reconsider the effects of their actions appear in the environment.

5. **Communication**. As interaction is a mandatory task for agents in MASs and Digital Ecosystems, the communication aspect should be well-defined. Communication is a means to express the interaction with the environment and ways to deal with exchange of messages between agents.

6. **Environment**. The system should be represented with its constituent elements; this enables to perceive the environment of the MAS-based Digital Ecosystems as a whole. Therefore, in order to create a model of an agent, these aspects should be taken into consideration so that an agent could meet the requirements to exist, survive, and benefit in MAS-based Digital Ecosystems.

As we have seen in the previous section, different methodologies have been proposed to assist the analysis and design of MASs. Most of these methodologies incorporate modeling languages for an agent concept. Following sections present representative works on agent concept modeling.

2.5.2 Agent Concept Representations in MASs

Modeling of agent concept is a prerequisite to properly design and develop MASs for various applications. Thus, there exist sound works on agent concept modeling in the literature. We present some of them as follows.

2.5.2.1 Conceptual Agent Model

In [1], the author proposes the entities of a conceptual agent as a Conceptual Agent Model framework (CAM). The constructs of a conceptual agent in CAM are categorized in three models to describe agent behaviors, namely: (i) Static Model, (ii) Dynamic Model, and (iii) Interaction Model. The Static Model describes the structural components of a conceptual agent and their relationships. The Dynamic model provides concepts to represent agent behavior. The Interaction model describes how agents interact with each other in a domain.

Before defining the constructs of a conceptual agent, this work gives an overview about the environment in which the agent is situated. The entity term is used to represent things in this environment. The attributes of these things can be defined as the state of the entity. Two kinds of entities are defined: dynamic and static. Dynamic entities have the ability to change the world and can perform actions which change the states of entities, while static entities do not have capabilities and can not perform actions. Dynamic entities also have rules which govern actions. According to this work, the agent itself is a dynamic entity which has beliefs and perceptions. There are also desired states which describe states of an agent, such as wants and goals. Agents also own compelling actions (such as perceiving, learning, and reasoning).

An agent is aware of the world through its perceptions and can affect its world by taking actions using resources. An agent has a capability to use these resources properly and performs actions to achieve a specific goal. It decides, using reasoning, which actions to take to achieve its goal. An agent observes its world and may form beliefs about the world based on its perceptions. By learning about the world in this way, the agent can then reason as to what it is going to do. An agent develops options of what it wants to do. These wants can be grouped together as a procedure and tell us what the agent wants to do to achieve its goals. Figure 2.1 shows the main constructs of to model an agent: Capability, Goal, Belief, Perception, Want, Procedure, and Action (Learning, Reasoning, Perceiving).

This work demonstrates an agent concept in a better way from its structure, behavior and interaction. However, this work mentions nothing about the role of an agent in resource and service provision and consumption. Agents might be organized according to their interested community or domain. In such case, the role of an agent becomes more important. Furthermore, this work lacks the necessary interface and means to support its communication and interaction.

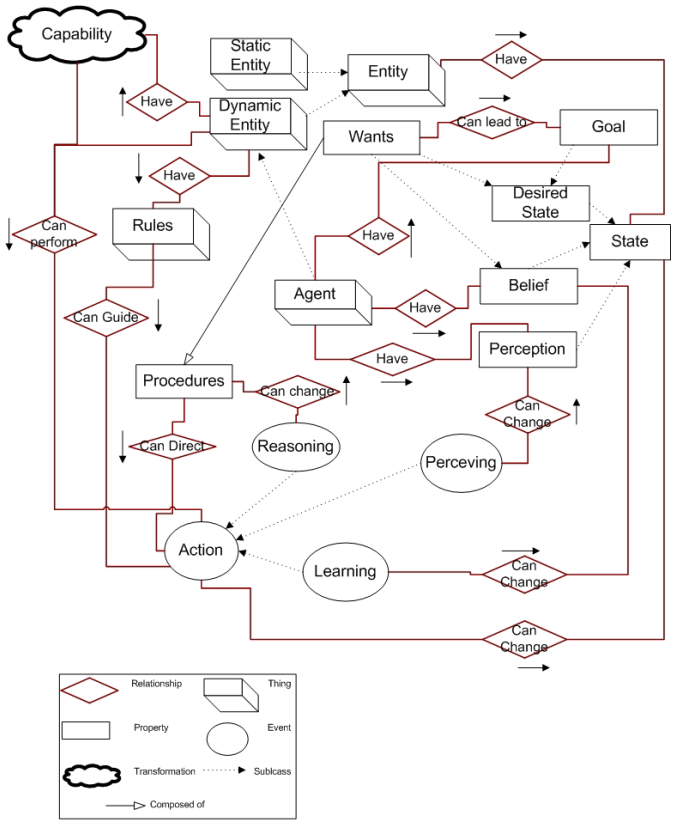

FIGURE 2.1: CAM Model [1]

2.5.2.2 Architecture Description Language Concepts for Agent

Based on the Belief-Desire-Intention (BDI)[4] Model [95], Architecture Description Language (ADL) concepts for specifying MAS architectures are discussed in [2, 96]. Authors introduce an ontological basis aimed at capturing a set of structural and behavioral concepts and their relationships. The main concepts are categorized in two models as shown in Figure 2.2: internal and global. The internal model captures the mental states of the agent and its potential behavior. Concepts like knowledge, capability, event, plan, and action are specified in this model. The agent needs to know about the environment in order to take decisions. This knowledge comprises a set of beliefs the agent has about the environment and a set of goals it follows. The intentional behavior of an agent is represented by its ability to react to events. An event can be generated by an action of an agent or by services provided by another agent. A plan defines the sequence of actions chosen by the agent to accomplish a task or to achieve a goal. The second

[4]The BDI model uses the Belief, Desire, and Intention concepts that are used correspondingly to symbolize and model an agent information state (what agent knows about itself and its environment), motivational state (what agent tries to achieve), and deliberative state (a plan to achieve agent's desired state of affairs).

model, global model, presents the interaction among agents that compose the MAS. This model provides an interface, which is composed of effector and sensor to provide or request services. A service is an action involving an interaction among agents. To recap, the internal/behavioral model is composed of nine entities: agent, knowledge base, goal, capability, belief, plan, event, action, and service; whereas the global/structural model is composed of the following entities: agent, configuration, architecture, interface, effector, sensor, and service. The second model demonstrates the whole picture of the MAS. The quality of this work is to clearly display the concepts at individual agent level and also as a whole while interacting each other to bear MASs. Despite its merit, this model misses considering the role and rule definitions of an agent.

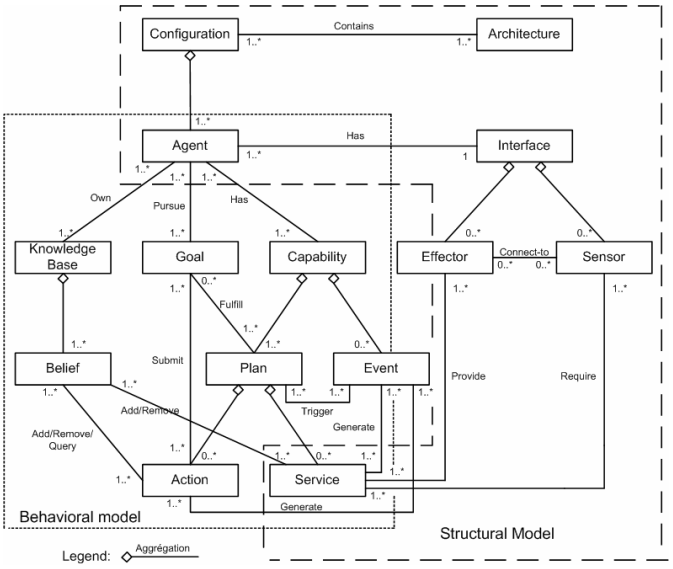

FIGURE 2.2: ADL Model [2]

2.5.2.3 Agent Model for Ontology-Driven Procedural Reasoning System

Authors in [3] propose an Ontology Driven-Procedural Reasoning System (O-PRS) agent model. In PRS-like systems, an agent is equipped with a list of plans. The main task of a PRS-like system, is to select and execute plans which are essential to achieve a goal or to respond to an event. Plan is a set of actions that an agent can perform and are selected based on the capabilities of the agent. PRS substitutes the abstract notions of desires and intentions in BDI to the more concrete concepts of goals and plans. The proposed O-PRS agent model uses ontological approach to represent the basic knowledge that PRS-like agents need to act as displayed in Figure 2.3, which are Believes, Plans,

and Events. Each agent may have one or more believes which may be corresponded to certain events and plans. When an event occurs for the agent, the corresponding plan for the same believe will be executed. However, this work does not clearly provide all elements to define agent structure, role, and rule definitions; and it provides nothing about how agents interact and communicate with each other.

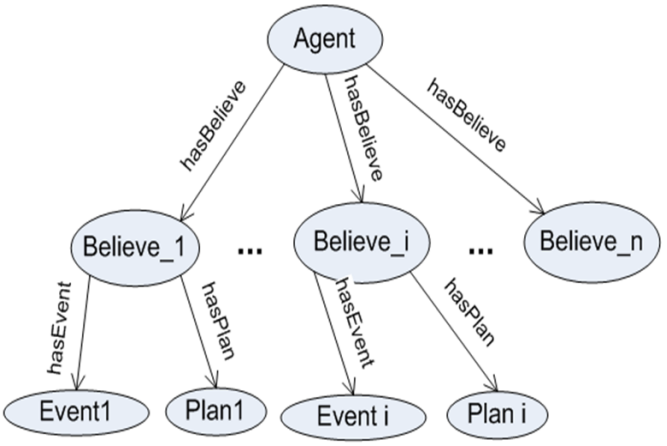

FIGURE 2.3: O-PRS Model [3]

2.5.2.4 Emotional Belief-Desire-Intention Model

In the work presented in [97], the authors describe an emotional agent model by combining concepts of BDI model with emotional aspects of an agent. Decision making process usually focuses on cognitive aspects of human performance. However, the emotional aspect is rarely addressed. Thus, this work considers emotion in agent modeling. Then, agents are built based on this assumption, would have both cognitive and reactive behavior. Following this, an agent could make reasoning based on its beliefs, desires, intentions, and emotions. Authors try to show the agent control loop, the emotional mechanisms, the resource usage, and the personality influence on agent behavior. Thus, to achieve their objective, the following concepts are combined in this work: Percepts, Beliefs, Desires, Options, Intentions, Emotions, Personality, and Resources. Percepts represent anything that comes from the environment. Beliefs are acquired from percepts. Desires are goals achieved by the agent. Options are alternatives to accomplish the desires which are generated based on current beliefs and intentions. Intentions are options that the agent has committed to. Emotions represent instinct behaviors of the agent. Emotions may influence percepts, beliefs, and options of the agent. Personality states the set of characteristics or emotional qualities that make the agent different form

others. And at last, resource concept is mentioned as it has a central role in decision making of the agent. According to this work, emotions influence the available resources the agent can use. Like most of works, agent role and interaction aspects have not been addressed.

2.5.2.5 ANote Model

ANote is a modeling language to offer a standard way to describe concepts related to the agent-oriented modeling process [98]. In [4], authors propose a development framework for building agent-based systems. As part of this work, a conceptual meta-model using ANote diagram is provided. This model has elements involved at high level and level of individual agents. As displayed in Figure 2.4, Concepts of this model includes, such as goals, interaction protocols, environment, resources, and organizations are defined at high level. Actions, communication, and plans are concepts that characterize individual agents. Having these two levels, ANote defines seven views based on its conceptual meta-model: goal, agent, scenario, planning, interaction, environmental, and organizational views. Goal view provides an initial identification of a tree of goals that outline the system functions. An agent view specifies the agent classes and agent classes specify the roles that will perform the goals elicited in the goal view. A scenario view captures agent behavior that shows how goals are achieved by agents. A planning view shows the agent's internal actions and their sequence. An interaction view is used to represent the set of messages agents exchange while executing an action plan. In fact, this work gives more emphasis to interaction and external environment. Less attention is paid to representation of agent structure and reasoning concepts.

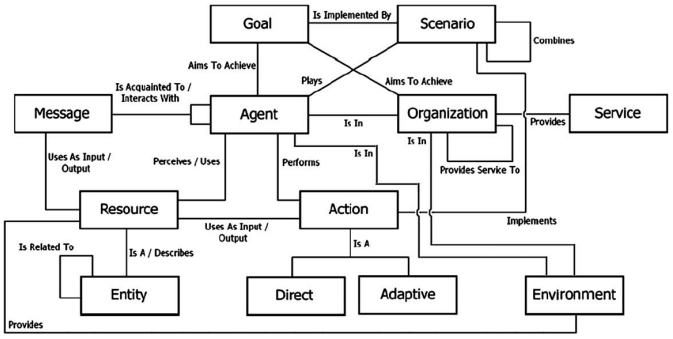

FIGURE 2.4: ANote Model [4]

2.5.2.6 MESSAGE Concepts

Authors in [5] provide a meta-model called MESSAGE to describe the agent concept elements as presented in Figure 2.5. Concepts are grouped into three categories : Concrete, Activity, and Mental-State. Concepts like Agent, Organization, Role, and Resources are under the category of concrete entities. An agent is an atomic autonomous entity that is capable of performing functions. A group of agents working together to a common purpose is referred as organization. A role is what an agent plays in this organization. Resource represents non-autonomous entities such as databases or external programs used by agents. The second category of entities in this model is called activity entities. Activity entities include concepts like: Task, Interaction, and Interaction Protocol. A Task is an activity performed by an agent. Interaction and Interaction Protocol define the way how agents communicate each other. Goal is the only concept mentioned as a type of mental entity. A Goal associates an agent with a state. In addition, two concepts are also used in MESSAGE to show an agent: Information Entity and Message. Information Entity represents an object encapsulating a chink of information which an agent perceives and a Message is an object communicated among agents. MESSAGE also defines a number of views that focus on overlapping sub-sets of entity and relationship concepts from its meta-model. The views (Organization, Goal/Task, Agent/Role, Interaction, and Domain views) are important to understand an agent concept from different directions and well understand MAS conception. However, this work does not provide internal structure and behavior of an agent and totally neglects the need of rule for reasoning and strategies to achieve its goals.

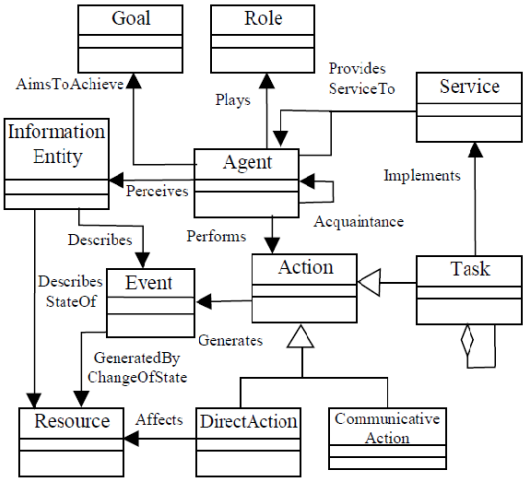

FIGURE 2.5: MESSAGE Model [5]

2.5.2.7 INGENIAS Model

In [6], the authors discuss a methodology and a set of tools for development of MAS called INGENIAS. IGNENIAS extends MESSAGE views by adding the environment view. Figure 2.6 shows views of this model. The environment view defines the entities with which the MAS interacts like resources, applications, and agents from other organizations. From different views present in this work, agent model provides the following concepts to represent an agent: task, goal, mental sate, and role. Except considering the environment of agents and MAS, this work shares the same limitation of MESSAGE.

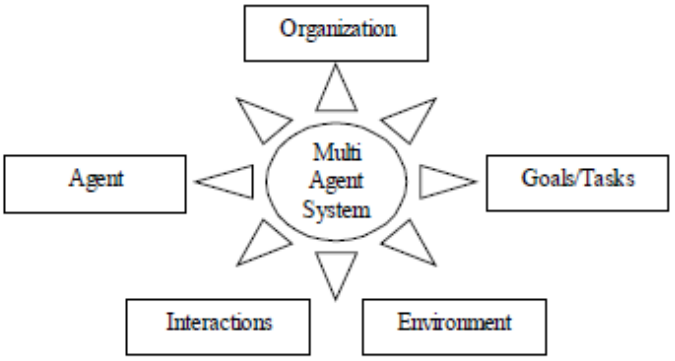

FIGURE 2.6: INGENIAS Model Views [6]

2.5.2.8 Agent concept in Ontology-based Modelling of MAS

In [7], the authors provide a conceptual view for MAS modeling from three main dimensions as shown in Figure 2.7: agents, environment, and organizations. This work specifies agents with their characteristics and roles in the organization. The agent concept is further described in terms of: Percept, Action, Belief, Message, and Plan. The organizational view indicates the role an agent could play in a group to accomplish its mission and achieve a goal. Besides graphical display of the three views, details about the various concepts are not provided in this work.

2.5.3 Agent Concept Representation in Digital Ecosystems

Authors in [8] introduce the concepts of Digital Ecosystem and its components: species and environment. Species is an individual or organization participating in the ecosystem which comes from certain domain, plays roles, and follows rules. It is driven by own profit

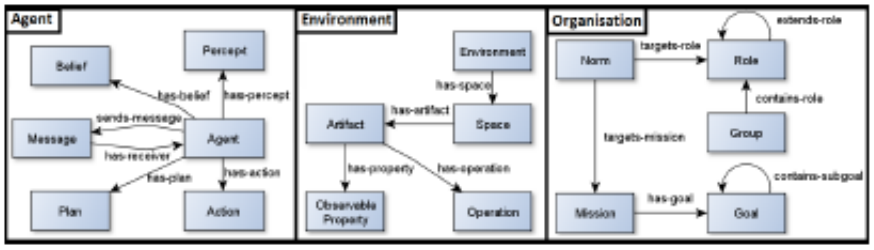

FIGURE 2.7: Main Concepts of MAS ontology model [7]

and carries out tasks that relate to the profit. Hence, species ontology is presented with the combination of elements as shown in Figure 3.3 i.e., Domain, Task, Profit, Rule, and Role (Supplier (Available Service) and Requestor (Requested Service)). Environment is the second constituent of a Digital Ecosystem which supports species. Thus, the Digital Ecosystem ontology has species and environment as sub components with little specifications of those components. In fact, this is the only work which provides a conceptual model of a Digital Ecosystem. Despite its contribution to model Digital Ecosystem, the internal structure and behavioral states of an agent are disregarded in this model.

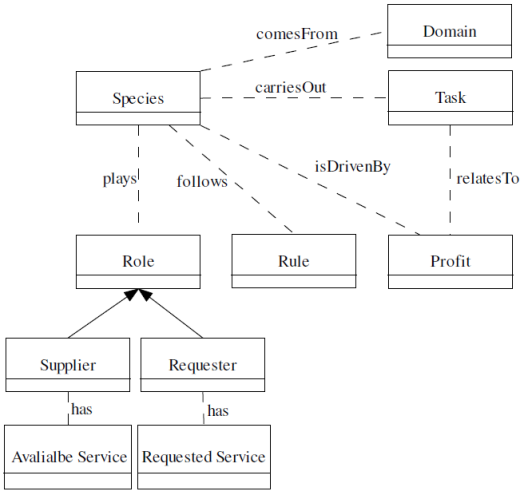

FIGURE 2.8: Species Ontology of Digital Ecosystems [8]

2.5.4 Discussion

Table 2.1 compares the models described before, with respect to our seven identified aspects of agent concept. Both the structural and behavioral aspects of agent enable interoperability among agents. More importantly this would help agents to express their

TABLE 2.1: Summary of Agent Concept Models

	Structure	Behavior	Role	Reasoning	Rule	Comm.	Env.
CAM [1]	Yes	Yes	No	Partial	No	Yes	No
ADL [96]	No	Yes	No	Partial	No	Yes	Partial
O-PRS [3]	No	Yes	No	No	No	No	No
EAM [97]	No	Yes	No	Partial	No	No	No
ANote [4]	No	Yes	Yes	No	No	Yes	Yes
MESSAGE [5]	No	Yes	Yes	Partial	No	Yes	Yes
INGENIAS [6]	No	Yes	Yes	Partial	No	Yes	Yes
Onto for MAS [7]	No	Yes	Yes	No	No	No	Yes
DES [8]	Yes	No	Yes	No	Yes	Partial	Partial

behaviors. However, most of the existing works focused on representing the internal state (behavior) in a form of belief, plan, and action. The reasoning/strategy aspect of an agent makes it intelligent in its decision making by properly analyzing the existing knowledge; whereas very few works give attention to this. Agents might be required to define and follow some constraints/rules while they are interacting with others. However, existing works forgot this aspect except one work [8]. Role definition is very important as agents exist as a community and all agents could not play the same role; only half of the reviewed works address it. Since, communication is vital for efficient agents interaction, majority of the works tries to address it though in different ways. The last one is the environmental aspect. In fact, the environment is not part of the agent rather it is a space where an agent exists. The way the environment is represented has an impact on agent modeling in its communication. Thus, considering this fact, most of the works address this.

To sum up, little attention has been given to the structure, reasoning, role, and rules. Moreover, the representation of each model of the agent varies from work to work. Many of the works defined the agent concept in different ways which shows that there is no comprehensive agent concept model. Therefore, it recalls a need to come up with a comprehensive and generic agent concept model that will support interoperability, facilitates communication, expressing agent behaviors for performing their actions in a strategic way, and make defining and following some constraints/rules.

2.6 Conclusion

Many researchers have been working on Digital Ecosystems in the last decade. They come up with different definitions, concept representations, and models/architectures about this newly introduced collaborative environment. In Digital Ecosystems, species can be digital, biological (e.g., human actors), and economic (e.g., organizations). These species can be represented in a form of software agents. Even though, there are various modeling languages for agents, there is still a need for more general conceptual models in the context of Digital Ecosystems to represent the environment and agents interaction, behavior, roles, rules, etc. Hence, a comprehensive and a generic agent concept modeling is required to properly design the entities of a Digital Ecosystem. The more an agent model is complete and clear, the easier is the definition of communication, coordination, and exchange mechanisms of agents in Digital Ecosystems. One of the contributions of this thesis is provide an ontological model considering the essential static and dynamic aspects of MASs by a clear representation of their concepts and relationships to support the design and development of MAS-based Digital Ecosystems.

Agent and multi-agent technologies have promised to address the characteristics mentioned in the Digital Ecosystem arena. With this, there are substantial works on MAS methodologies to help developers. Current modelling languages and methodologies that support the construction of such systems require the use of different tools to complete design, development, and deployment processes. These methodologies vary in their scope, approach, modelling concepts, and intended purposes. Despite the current efforts provided in the literature, the development of MASs remains a complicated task, which demands time and special programming skills. Most of them focused on designing and integrating ontologies in defining the domain of the system or to represent agents knowledge. However, there is no work which considers ontologies in the whole development process of a MAS. This is the hole that our study intends to address to enable developers to create MAS-based Digital Ecosystems in a simple and quick way from ontology model. It is, therefore, crucial to propose a framework how to design and develop such systems from the ontological representation of agents. Thus, the main objective of this work is to provide a framework for designing and generating a Digital Ecosystem of any kind from an ontological model in order to reduce the development effort.

Chapter 3

Ontology for MAS-Based Digital Ecosystems

3.1 Introduction

Multi-Agent Systems (MASs) have become one of the most promising technologies exploited in several complex applications, especially in collaborative systems to increase the efficiency and effectiveness of working groups in a distributed environment [82, 99]. Nowadays, there is a wide range of existing application domains that are making use of agents and develop MASs to solve larger and more complex problems. MASs are complex systems that integrate a collection of autonomous agents that have their own goals and actions and are able to interact, collaborate, and exchange knowledge [100]. Digital Ecosystem is one of the areas in which MASs are appropriated as a means for modeling and simulating such complex systems [3]. Digital Ecosystems provide the basis for collaborative environment where agents interact with each other to reach their individual or shared goals. Agents in Digital Ecosystems are self-organized, interactive, reactive, and proactive. Consequently, they are being capable of acting autonomously, making decisions, and fulfilling responsibilities. Our motivation is the development of Digital Ecosystems for multimedia sharing and collaboration from the basis of MASs.

Ontology has been commonly used to specify the domain for a MAS and knowledge for individual agents to support their communication and coordination. For agents to uniformly interpret the exchanged messages and generate new knowledge, they need to

share the same understanding of the concepts conveyed in the messages [7, 101]. Ontologies are rich and highly expressive knowledge models that are used to specify the semantics of the message and to actually represent the knowledge [101]. Ontologies enable agents to fully understand the knowledge domain and to identify possible links between different ontology concepts. Without the mutual understanding that an ontology provides, the knowledge being passed might be misinterpreted by one of the agents. This is one of the reasons presented in [102] for the use of ontologies in a agent-based systems. Thus, agents could represent their knowledge in a standardized and consistent way without any ambiguity [103].

Thus, for Digital Ecosystems, additionally to ontologies to represent the shared knowledge under a specific domain, there is still a need for more general conceptual models and methodologies to represent the specific characteristics of Digital Ecosystems in terms of win-win interaction, engagement, equilibrium, and self-organization. In the context of multimedia contents, there exist in the literature different ontologies that can be used to represent this specific domain [104]. However, a common and shared vocabulary is also required to define the environment and organizations of the ecosystem, as well as the communication ways, structure, behaviors, roles, and rules of agents in the Digital Ecosystem. Ontologies can be used for MAS-based Digital Ecosystems modeling and development, considering the simplicity, efficiency, and speed in code generation, that it offers to develop MASs and Digital Ecosystems. The aim is to provide the essential static and dynamic aspects of the Digital Ecosystem by a clear representation of their concepts and relationships. In this case, ontologies will be a means to design and develop MAS-based Digital Ecosystems.

While there are a large variety of works proposing methodologies, tools, approaches, strategies to integrate ontologies to MASs for knowledge representation and sharing under specific domains, few (or none) of those works incorporate ontologies to support the development itself of MASs, independently of specific domains [105]. Moreover, there is no any previous work that uses an ontology in the global modeling of MASs as a conceptual representation to instantiate the system. This gap in the relationship between ontologies and MASs gave us the chance to investigate how a general ontology can be used for MASs modeling and development. It is known that the development of MASs demands a considerable amount of time and high level programming skills [106]. Therefore, our aim in this work is to present MAS2DES-Onto, an ontological model, to

provide all the essential aspects of MASs by a clear representation of their concepts and relationships to support the design and development of MAS-based Digital Ecosystems. The next sections discuss the requirements and concepts of the proposed ontology in detail.

3.2 Modeling Requirements for Multi-Agent Systems and Digital Ecosystems

The goal of MAS2DES-Onto is to address the requirements of MASs and Digital Ecosystems in one conceptual model so that a MAS could be used as a means for the design and development of a Digital Ecosystem. Thus, we first discuss the requirements of the two systems.

3.2.1 MAS Modeling Requirements

A MAS consists of a group of agents that can potentially interact with each other [107]. MAS environment provides an infrastructure specifying communication and interaction protocols. This environment is typically open and has no global control; and contains agents that are autonomous and distributed. These agents may be self-interested or cooperative.

The most important characteristics of agents that form a strong basis for the design of highly effective and efficient MASs are [108, 109]:

- **Autonomy**. An agent needs to be autonomous; it needs to be independent when making decisions and performing actions. It demands to be able to make appropriate choices and satisfy its design objectives. An agent inquiries to be able to make rational decisions based upon the available information, but also to determine what to do after an action either succeeds or fails.
- **Sociability**. An agent needs to show social abilities by interacting and communicating with other agents especially in MASs where the agents are collaboratively working towards a shared goal. This is achieved by engaging in social activities like cooperation, coordination, and negotiation.

- **Reactive**. An agent needs to be reactive by perceiving their environment, and recognize and understand changes in that environment. Agents need to know how to react to these changes and respond to them in a timely manner.

- **Proactive**. An agent needs to be able to initiate actions that will help to achieve its goals. This means that an agent needs to appreciate the state of its environment and decide how best to fulfill its target mission.

Agent modeling criteria can be derived from the characteristics of an agent and its internal data structure. Autonomy, social ability, reactivity, and pro-activeness of agents can be used as a set of design criteria for MAS [109]. The second aspect of agent's design criteria is considered from its internal data structure [101]. It is a major factor in the decision making process of agents. This contains domain knowledge description, knowledge about the agent's environment and the agent's history files. An agent needs to be familiar with its environment in order to function efficiently. It also needs to have information about its previous experiences for the purpose of improving itself and increasing efficiency.

3.2.2 Digital Ecosystem Requirements

The requirements for modeling a Digital Ecosystem are derived from its basic essences and characteristics [20]:

- **Interaction and engagement**. Agents should need to interact and engage within each other to find interesting things and to share resources. Interaction and cooperation depend on preferences and interests of participants and the collaboration is mostly based on agents mutual benefit.

- **Self-organization**. It signifies that each agent is independent and self-empowered to undertake actions and decisions but also agents are coming to this environment to collaborate with other agents for mutual benefit. With this, agents in Digital Ecosystems must be ensured of resource ownership and control.

- **Balance**. This is to keep the benefits of all agents in a mutual way and act responsibly for the safety and sustainability of its environment.

- **Demand-driven**. This requirement is to show that agents come to the Digital Ecosystems by their own choice. Agents can also determine their own requirements and interests.
- **Role**. Each participating agent can play dual roles of providing and consuming resources and services as there is no fixed role in Digital Ecosystems.
- **Domain-clustered**. This requirement states that species join the Digital Ecosystem environment where they have something in common or share the same interests.
- **Heterogeneity agents in a Digital Ecosystem**. This is also an important requirement as Digital Ecosystems encompass moral and digital agents.

Taking into account these two group of requirements, next we illustrate the main components of MAS-based Digital Ecosystem in the context of a motivating scenario.

3.3 MAS-based Digital Ecosystem for Documentary Movie Production

From the motivating scenario provided in Chapter 1, we illustrate in this section, how the requirements of the documentary movie producer can be addressed from the basis of MAS. The proposed MAS must take into account several issues, such as: (i) how to deal with different information sources and to select and retrieve information from those sources; (ii) how to suitably verify, clean, classify, and categorize collected data in order to put into evidence the content useful to discriminate among categories; this is how to get the relevant documents and properly classify them for proper utilization in the process of making the documentary movie; (iii) how to aggregate the classified data to generate content from multiple sources; (iv) how to extract knowledge from the aggregated data for analysis and use the discovered knowledge to aggregate the information semantically from different location, services; and (v) how to exploit the expertise of others so as to produce the documentary with minimal effort.

To address the above issues, we propose a MAS that has four tasks and a coordinator. List of agents is identified to run the tasks in organized manner. The tasks with the necessary agents are described as follows:

1) Preprocessing Task

This task requires two agents that are entrusted with collecting and extracting data and services from the potential sources to filtering and encoding tasks. Information sources are mainly social media sites and databases of different stakeholders of the domain of Ebola disease. Information is retrieved by suitable agents, devised to extract the preferred data from various sources. Agents of this level are aimed at encoding (verify, clean, etc.) the information extracted and filtering it to retaining only relevant data. The actual encoding strictly depends on the specific application (preprocessing activities, such as feature selection could be considered to prepare the data to be processed). Agents for this task are: Discovery and Filtering Agents.

- **Discovery Agent**. The task of this agent is to search and collect relevant data (e.g., photos and videos of infected people) for producing the required documentary. This agent does discovery and selection of data from social media sources. Once collected, all the information is suitable encoded to facilitate the preprocessing task. This agent sends the retrieved information to the filter agents.
- **Filtering Agent**. This agent prepares data for classification and analysis tasks (e.g., types and forms of data). It is responsible for performing the necessary data cleansing and verifying before using the data set for making the documentary movie.

2) Processing Task

The agents ares responsible for classifying and clustering, and extracting knowledge from the clustered data for analysis purpose. Classifier and semantic extractor are the agents serving in this module.

- **Categorizer Agent**. This agent is responsible for classifying and categorizing the collected and filtered data from various social media (e.g., per country, date, or gender).
- **Semantic Extractor Agent**. This agent does extraction of knowledge from the categorized data. And based on the extracted knowledge, aggregation of data will be done for analysis purpose. This agent generates contents which go through tasks like integrate, merge, split, and the like.

3) Analysis Task

The agents play the role to analyze the outcome from the processing module based on some specific strategies and actions in supporting the user's decision making. Integrator and Knowledge Manager agents are identified to run the analysis task.

- **Integrator Agent**. It is an intelligent agent that analyzes information, operates valuable information to find the relationship between different pieces of information to provide precautions patterns.
- **Knowledge Manager Agent**. This agent generates new knowledge from the results of the integrator agent and also provides new findings and instances to enrich the knowledge base and domain ontology of the system. It is mainly responsible for generating summary statements and reports on the thematic issue in order to detect major trend changes (e.g, statistics in age, gender,country, death rate).

4) Supporting Task

Agents are required to support Alice in different ways for manipulating, accessing, publishing, and sharing resources. Here we consider two agents for Alice to edit and interact with the system. Editor and Interface agents are named to support users like Alice.

- **Editor Agent**. After getting the necessary inputs, this agent can start editing. It adds items to the storyboard, and then trims and arranges items on it. This agent also enables to add titles, narration, soundtracks, transitions, and effects.
- **User Interface Agent**. This agent interacts with Alice (represented as user agent). It supports users the publication of the documentary film and share it with other people with minimal effort and expertise. This agent also enables users to set the source from which data will be extracted, and the topics she is interested in.

5) Coordinator Agent

This agent forms the core of the Alice MAS-based digital Ecosystem, which plays a role to coordinate various agents' activities. It is able to break a task into sub-tasks that are directed toward the appropriate agents, and then combines results from separated

agents for the overall response to a task. In addition, it is also responsible for create or add new agents, delete or suspend out-of-date agents and implemented techniques. Besides, the interactions and communications between agents should be handled by this agent.

The next section describes a conceptual model that represents basic concepts of MAS and digital Ecosystem, to help on the modeling and development of MAS-based Digital Ecosystems.

3.4 MAS2DES-Onto

The MAS development process basically focuses on two major aspects: individual agent and agent organization in view of agents' coordination and interaction as we show in the Alice scenario. Existing agent modeling languages are incomplete in handling all dimensions of an agent in the context of MASs and Digital Ecosystems. Some models are focused on the internal behavior of an agent whereas others are in favour of representing its structure. More details about the limitations of existing agent models can be found in Section 2.5. In general, the model of an agent varies from work to work and there is no wall-to-wall agent concept model. Additionally, there is no MAS model that takes into account Digital Ecosystems. Hence, this urges to come up with a comprehensive agent model that supports the development of MAS-based Digital Ecosystems.

Accordingly, we propose, MAS2DES-Onto, a meta-model developed to offer a standard way to describe concepts related to agents in order to support the MAS development process in the context of Digital Ecosystems. This generic ontology is designed to address the requirements of MASs and Digital Ecosystems. Figure 3.1 provides the core concepts of MAS2DES-Onto. This meta-model defines, at the high level, the interactive and environmental (system) level concepts. At the level of individual agents, it has elements to represent basic agent concepts from its internal and external side. These concepts define a variety of different aspects or views, which may complement each other during the system development stages. MAS2DES-Onto integrates concepts to determine the elements to be used to specify a MAS and what should be present in a Digital Ecosystem. Hence, the purpose is to use it in support of MAS-based Digital Ecosystem design and development.

Before we use agents to model any kind of MAS-based systems, the constructs must be clearly defined and cover the static, dynamic, interactive elements of the conceptual agent and the environment. To do so, we plan to afford five conceptual modules (see Table 3.1): structural, species, reasoning, interaction, and system modules. The modules consist of collection of entities and instances of entities and relationship concepts. The following sub-sections describe further these modules.

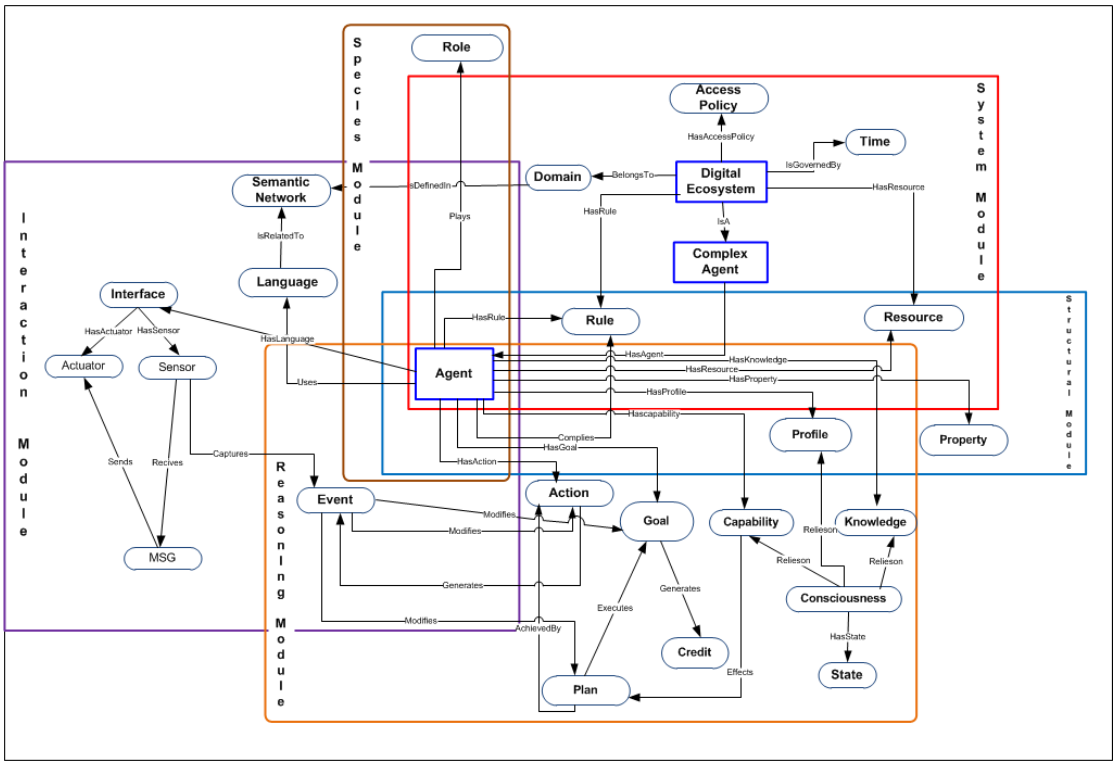

FIGURE 3.1: Conceptual Model for MAS-based Digital Ecosystem

TABLE 3.1: MAS2DES-Onto Modules

Modules	Description
Structrual	Represents the concepts describing agent structure in MAS and Digital Ecosystems (components, properties, etc.)
Species	Represents the concepts describing agent types and roles. This module is mainly to address requirements of Digital Ecosystems
Reasoning	Represents the concepts related to the agents reasoning and actions. This module is to meet requirements of MASs and Digital Ecosystems
Interaction	Represents the concepts allowing an agent to interact with others and the environment. This module is to meet requirements of MASs and Digital Ecosystems
System	Represents the concepts related to the environment. This module is the Digital Ecosystem environment

3.4.1 Structural Module

This module represents concepts describing basic components and properties of the agent structure (see Figure 3.2). It is composed of the following concepts: *agent*, *property*, *profile*, *resource*, and *rule*. An *agent* is an atomic autonomous entity that is capable of performing functions and represents participants in the system. A *property* describes attributes that characterizes an agent which can be either *common* to all agents or dedicated (*personal*) to a specific agent. A *profile* is a set of information describing an agent with its *preferences*. A *resource* is a concept which represents the set of assets that an agent possesses and manages. A *rule* defines condition or constraint set by a specific agent that others should follow while interacting. The detailed description of the main concepts in this module is available in Table 3.2. This module mainly provides the structure to fulfill the requirements of MASs. In addition, to meet the requirements of an agent in Digital Ecosystem environments, *profiles* and *rules* are incorporated. These concepts are needed to provide agent interests and *preferences* for better collaboration. *Rules* are essential to keep benefits of an agent in the Digital Ecosystem, while interacting with others.

To illustrate the concepts of this model, let's go back to Alice MAS-based Digital Ecosystem for documentary movie production. Alice as an example of user agent, she has a *profile* as a journalist with her *preferences* in Ebola disease and in the specified region and time. Alice comes to the system with her *resources* in hand about the disease; However, she is looking for more *resources* to produce a better documentary. The aim of Alice is to publish and share the documentary to those who are ready to respect her usage *rules*. As the documentary is focusing in the health sector, there are a lot of private issues (e.g., image of infected people). This representation of Alice is an important input to the Discover Agent in order to fetch data that meet her requirements.

3.4.2 Species Module

As illustrated in Figure 3.3, the species module consists of *agent* types with their *roles*. In Digital Ecosystems, we can have both *moral* and *digital agents* unlike MAS, which only has digital agents [108]. The *role* concept defines the part played by an *agent*. An *agent* may be capable of playing several *roles*, such as *consumer* (who sends *requests* to access

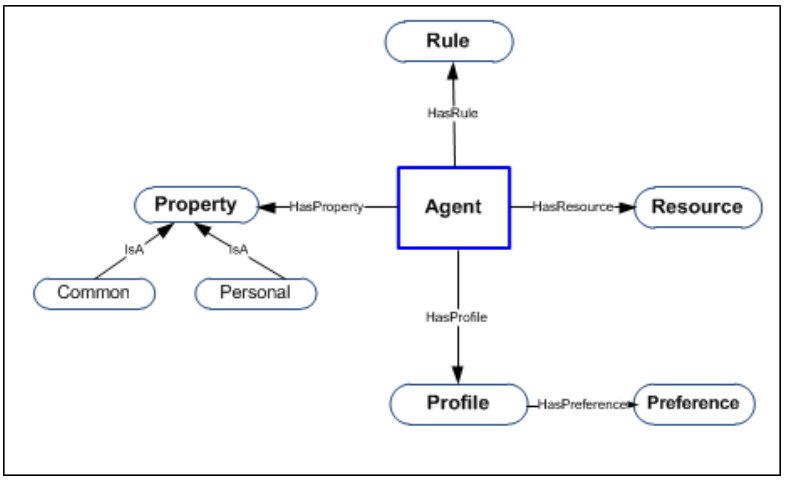

FIGURE 3.2: Structural Module of MAS2DES-Onto

TABLE 3.2: Structural Module Concepts Description

Structural Module	**Description**
Agent	Agent is the core concept which represents an atomic and an autonomous entity in the Digital Ecosystem. An agent is related to all the concepts in the rest of the modules of MAS2DES-Onto
Property	A property is a set of attributes that characterize an agent (MAS2DES-Onto:Agent) which can be either common (MAS2DES-Onto:Common) in all agents (e.g., location) or specific (MAS2DES-Onto:Personal) to individual agents
Common	A common property is a property (MAS2DES-Onto:Property) that can be found in all agents (MAS2DES-Onto:Agent) (e.g., an agent location, name)
Personal	A personal property is a property (MAS2DES-Onto:Property) dedicated to a specific agent (MAS2DES-Onto:Agent) (e.g., the storage unit of an information retrieval agent)
Profile	A profile is a set of information describing an agent (MAS2DES-Onto:Agent) such as interests and preferences (MAS2DES-Onto:Preference), which are taken into account during the reasoning process (thus, this concept is also in the Reasoning Module)
Preference	A preference is one attribute of a profile (MAS2DES-Onto:Profile) describing desire of an agent (MAS2DES-Onto:Agent), which is taken into account during the reasoning process (thus, this concept is also in the Reasoning Module)
Resource	A resource is a concept which represents the set of assets (e.g., data, processing time, processing unit, storage unit) that an agent (MAS2DES-Onto:Agent) owns and manages to fulfill a goal (MAS2DES-Onto:Goal). It is a subset of resources (MAS2DES-Onto:Resource) available at the Digital Ecosystem level (thus, it is also a concept in the System Module) and can be involved in agents (MAS2DES-Onto:Agent) interaction (thus, it is also a concept of the Interaction Module)

services/resources), *provider* (who sends *responses* to *requests*), and *orchestrator* (who coordinates some activities in the system). Also, multiple *agents* may be able to play the same *role* [110]. Table 3.3 displays detailed descriptions of these concepts. This module only addresses requirements of Digital Ecosystems. As there is no fixed *role* of agents in Digital Ecosystem, they may play different *roles* depending on the context. This is decisive for agents to determine to whom to communicate and set their expectations.

For instance, in the documentary movie production system, we have human being actors like Alice and other software components that are supporting movie makers (e.g., video camera, movie editor tools). This is a typical example that in Digital Ecosystems we could have *moral* and *digital* entities. These entities could play different types of *roles*. For example, Alice is a *consumer* who sends her request in looking for supportive resources for producing movie on Ebola. Not only this, but also different agents can be categorized as *producer* and *consumer* like Discovery Agent plays a *producer role* for Filter Agent as it surfs the web and forwards the collected resources for the next step of processing. In this way, Filter Agent plays a *consumer role*. At the same time, the result of Filter Agent is forwarded to Categorizer and Analyzer Agents for further processing. In this context, Filter Agent is a *producer* to agents of the next steps. As we said it before, in Digital Ecosystems agents may play both *roles*. The Coordinator Agent plays a role of orchestration by decomposing the overall task into smaller tasks and assigning these sub-tasks to the various agents.

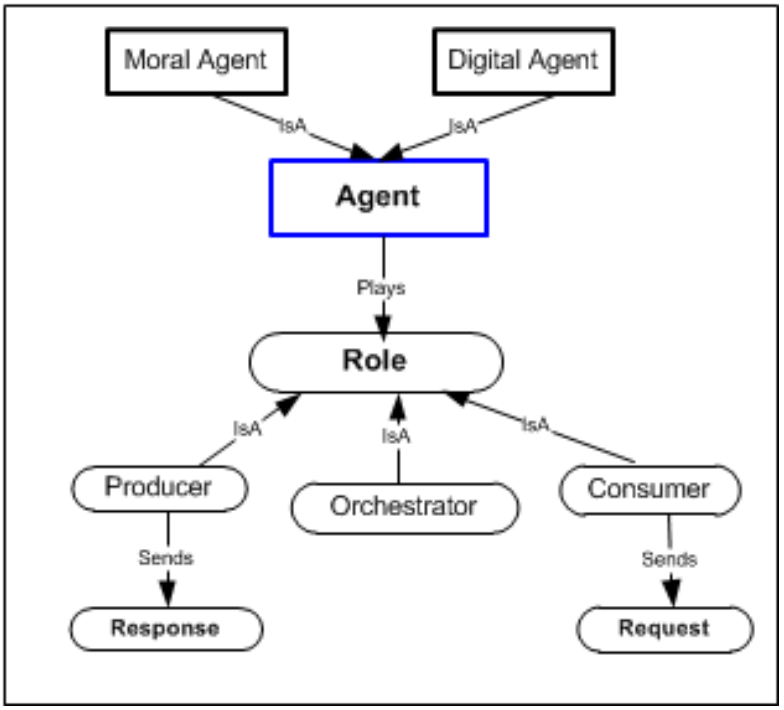

FIGURE 3.3: Species Module of MAS2DES-Onto

3.4.3 Reasoning Module

Reasoning is vital for agents to survive and act intelligently in the environment where they exist. An agent can make decisions based on its beliefs, desires, intentions (specified

TABLE 3.3: Species Module Concepts Description

Species Module	Description
Role	A role defines a duty an agent (MAS2DES-Onto:Agent) plays such as a provider (MAS2DES-Onto:Provider), consumer (MAS2DES-Onto:Consumer), or orchestrator (MAS2DES-Onto:Orchastrator)
Producer	A producer is a role (MAS2DES-Onto:Role) which refers to an agent (MAS2DES-Onto:Agent) that provides resources (MAS2DES-Onto:Resource) and services to others
Response	A response represents a reply/feedback sent by a producer agent (MAS2DES-Onto:Producer) for a request (MAS2DES-Onto:Request). Hence, it is also a concept of the Interaction Module
Consumer	A consumer is a role (MAS2DES-Onto:Role) which refers to an agent (MAS2DES-Onto:Agent) that sends a request to other agents
Request	A request represents a desire of a consumer agent (MAS2DES-Onto:Consumer) for querying a resource (MAS2DES-Onto:Resource). It is also a concept of the Interaction Module
Orchestrator	An orchestrator is a role (MAS2DES-Onto:Role) which refers to an agent (MAS2DES-Onto:Agent) that coordinates and manages agents
Digital Agent	A digital agent is an agent (MAS2DES-Onto:Agent) that represents a digital entity (e.g., a software application)
Moral Agent	A moral agent is an agent (MAS2DES-Onto:Agent) that represents a moral entity (e.g., human-being, company)

in its *profile* and *preferences*), and *capability*. Indeed, agents which are either in MASs or Digital Ecosystems are assumed to crown a characteristics of being proactive and reactive. Obviously, the level of expected reasoning skill in Digital Ecosystem agents must excel than agents in MAS. agents in Digital Ecosystem should react in a responsible manner for the safety and sustainability of its environment which requires high level of reasoning. The reasoning module, shown in Figure 3.4, consists of mental states of an agent and dependencies among them.

An agent needs *knowledge* about its environment in order to make good decisions. However, *knowledge* about the current *state* of the environment is not always enough to decide what to do. In other words, in addition to a current *state* description, the agent needs *goal* information. A *goal* represents aims and desires that an agent wants to achieve. *Goal* achievement requires commitment. Thus, to maximize the *credit* of success, an agent need to acquire a *capability*. The intention and ability of an agent is represented by its *capability* to react to its *plans*. A *plan* defines the sequence of *actions* or services to be chosen by the agent to accomplish a task or fulfill a *goal*. An *event* is generated either by an *action* or by services provided by another agent; which may modify *plan*, *action*, and *goal*. Agents must also comply *rules* which govern *actions* while communicating to others. *Consciousness* is cumulative effect that arises from agent *knowledge*, *capability*, and *profile*. These all concepts are important to have proactive and reactive

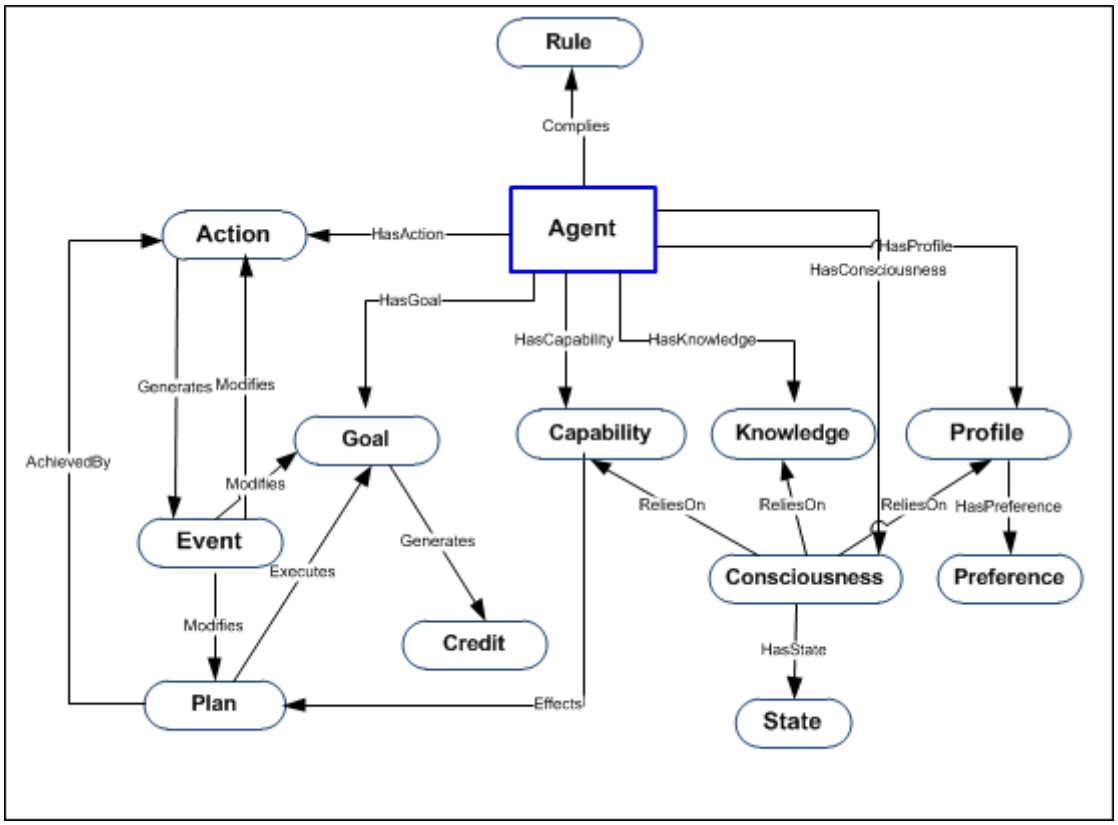

FIGURE 3.4: Reasoning Module of MAS2DES-Onto

agents in the system so that they can quickly deal with unexpected events [97]. The detailed description of the concepts are found in Table 3.4.

In the context of Alice MAS-based Digital Ecosystem for documentary movie production, we are considering Integrator Agent module to illustrate some of the concepts of Reasoning module. The goal of Integrator Agent is to analyze the extracted *knowledge* to find out facts, patterns, trends, etc. The analysis result will be incorporate in the documentary movie to make it more valuable for decision makers. To achieve this, it should have a *capability* of data mining, analysis, and providing results. In order to reach this, there are sequences of *actions* like: getting extracted knowledge, acquire analysis parameters, deduce patterns and trends, and offer analysis results. Thus, the Reasoning Module is designed to represent such aspects of agents.

3.4.4 Interaction Module

One of the basic essences of agents in MASs and Digital Ecosystems is interaction. Interaction serves as key element to support communication and the construction of the

TABLE 3.4: Reasoning Module Concepts Description

Reasoning Module	**Description**
Capability	A capability defines the agent's (MAS2DES-Onto:Agent) reasoning and learning competence to use properly resources (MAS2DES-Onto:Resource) and take appropriate actions (MAS2DES-Onto:Action) according to defined rules (MAS2DES-Onto:Rule). Without capabilities, the agent cannot reason
Goal	A goal is a desired result than an agent (MAS2DES-Onto:Agent) plans and commits to achieve. It is one of the messages communicated among agents as they may share the same goal (MAS2DES-Onto:Goal). Accordingly, the goal is also a concept of the Interaction Module
Credit	A credit represents accomplishment of an agent as a consequence of its goal (MAS2DES-Onto:Goal)
Plan	A plan is a capability (MAS2DES-Onto:Capability) denoting that an agent (MAS2DES-Onto:Agent) can have a sequence of actions (MAS2DES-Onto:Actions) in order to achieve a specific goal (MAS2DES-Onto:Goal). Agents might share and exchange plans in a form of messages. Thus, plan is also a concept of the Interaction Module
Action	An action defines operation of an agent (MAS2DES-Onto:Agent)
Knowledge	A knowledge represents what an agent (MAS2DES-Onto:Agent) knows about itself, other agents and the environment
State	A state represents a condition that an agent (MAS2DES-Onto:Agent) is in at a specific time (e.g., Idle state, Active state)
Consciousness	A consciousness quantifies the state of awareness of an agent (MAS2DES-Onto:Agent) about the environment within and around depending on its reasoning capabilities (MAS2DES-Onto:Capability), knowledge (MAS2DES-Onto:Knowledge), profile (MAS2DES-Onto:Profile), and current state (MAS2DES-Onto:State)
Event	An event represents an occurrence happening to an agent at a determinable time. It may affect actions (MAS2DES-Onto:Action), goals (MAS2DES-Onto:Goal), and plans (MAS2DES-Onto:Plan) of an agent (MAS2DES-Onto:Agent). Events are captured by a sensor (MAS2DES-Onto:Sensor) of an interface (MAS2DES-Onto:Interface). Hence, this concept is also in the Interaction Module

system [111]. Figure 3.5 displays the concepts of this module. An agent interacts with its environment through an *interface*. An *interface* is a specification of how an agent appears to the rest of the system. It is composed of a set of *actuators* and *sensors*. An *actuator* provides a service that is available to other agents, and each *sensor* requires a service from another agent and captures *events*. To effect this, a *message* (*MSG*) is the means; which is an object communicated between agents. The correspondence between a *request* and a *response* defines an interaction through a *message* (*MSG*). Agents can interact by asking for or sharing *goals*, *plans*, *resources*, and semantic meanings (*terms*). Agents in Digital Ecosystems are heterogeneous and domain-clustered. In order to support and bring semantic interoperability among agents, common communication vocabulary (*language*) must be defined with their semantic relationships (*semantic network*). All these concepts are detailed in Table 3.5.

To illustrate more the above description, we describe the User Interface Agent of MAS-based Digital Ecosystem for documentary movie. This agent is to assist Alice in forming

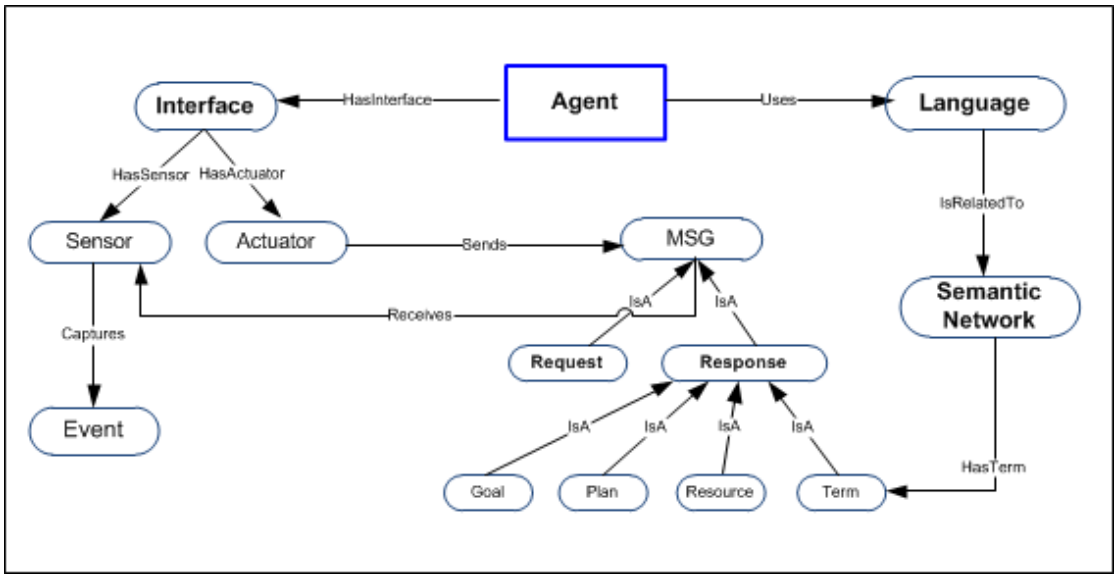

FIGURE 3.5: Interaction Module of MAS2DES-Onto

TABLE 3.5: Interaction Module Concepts Description

Interaction Module	Description
Semantic Network	A semantic network is a representation of semantic relations between concepts which are used as a form of knowledge (MAS2DES-Onto:Knowledge) representation of a domain (MAS2DES-Onto:Domain). It is a graph, where vertices represent concepts and where edges represent relations between concepts
Term	A term is a word or a phrase used to express a concept in a semantic network (MAS2DES-Onto:SematicNetwork)
Language	A language is a communication vocabulary for agents (MAS2DES-Onto:Agent) to understand each other and is related to a semantic network (MAS2DES-Onto:SemanticNetwork)
MSG	A MSG is a request (MAS2DES-Onto:Request) or a response (MAS2DES-Onto:Response) communicated between agents (MAS2DES-Onto:Agent). It may convey agent's goal (MAS2DES-Onto:Goal), plan (MAS2DES-Onto:Plan), resource (MAS2DES-Onto:Resource), and terms (MAS2DES-Onto:Term) among agents. It is sent and received respectively by actuators (MAS2DES-Onto:Actuator) and sensors (MAS2DES-Onto:Sensor)
Interface	An interface is the front-end of an agent (MAS2DES-Onto:Agent). An agent (MAS2DES-Onto:Agent) interacts with the environments and other agents (MAS2DES-Onto:Agent) through its Interface
Sensor	A sensor is a component of an agent's interface (MAS2DES-Onto:Interface) that enables the agent (MAS2DES-Onto:Agent) to receive messages (MAS2DES-Onto:MSG)
Actuator	An actuator is a component of agent's interface (MAS2DES-Onto:Interface) that enables the agent (MAS2DES-Onto:Agent) to send messages (MAS2DES-Onto:MSG)

queries (e.g., all documents of Ebola in Sierra Leone) as well as to present the retrieved and assembled information back to Alice. Thus, User Interface Agent communicates user's request to the Coordinator Agent. This agent receives user queries in a form of *message* and also sends response and feed-backs to users. To accomplish this, it should have an *interface* that has a *sensor* and an *effector*. In fact, all the agents in the documentary movie production have an interface as the agents are interacting each other to accept inputs and deliver outputs.

3.4.5 System Module

The system module essentially specifies concepts that compose a Digital Ecosystem environment. Only agents are found in MAS environments, whereas a Digital Ecosystem environment has agents and other important elements that facilitate sharing and collaboration among participating agents. As it is shown in Figure 3.6, a *Digital Ecosystem* concept consists of *agents*, *complex agents*, *resource*, *rule*, *time*, *access policy*, and *domain*. An *agent* represents a participant in the *digital ecosystem*. *Complex agents* represents an agent which consists of other agents under it. In Digital Ecosystem, all agents will come to the environment with some *resources* and should obey the system level *rules*. These agents are also free to define their resource sharing and usage policy (*access policy*). Every *Digital Ecosystem* is assumed to have one or more application *domain* in a given period of *time*. Table 3.6 details concepts at system level.

According to Alice scenario, this system has different kinds of agents (human and digital). Each of the agents has resources that help the movie production. Of course, some are providers and others are consumers. Each of these entities should respect the usage policy of a documentary movie maker (like Alice) and Alice also should respect others rule while she tries to manipulate data of others. The domain is clear that it is health as the documentary is focusing on Ebola virus. Time is another important factor in dynamic situations like the Ebola epidemic. For example, the Coordinator Agent waits for all the Discover Agents to supply the results or until the deadline is reached. Otherwise, the user might not get the required resources on time. In addition, the source of data (like social media) could evolve and the vital data might be removed from storage or replaced by others. Finally, Alice, the movie maker, is looking for an environment for publishing and sharing the documentary movie in a way that keeps her usage rule. Thus, the system should allow her to define different usage rules so that she can make available the final movie to those who respect her access policy (e.g., Alice can share her movie only to organizations working in Ebola).

3.4.6 MAS2DES-Onto Relationships

MAS2DES-Onto conceptual models provide not only the definition of concepts but also relationships among those concepts. Two types of relationships are identified.

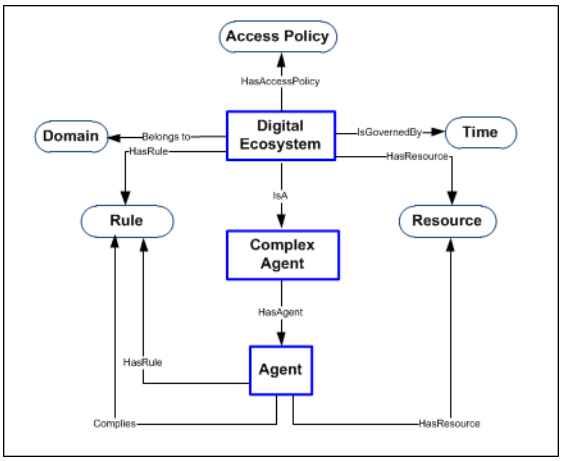

FIGURE 3.6: System Module of MAS2DES-Onto

TABLE 3.6: System Module Concepts Description

System Module	Description
Digital Ecosystem	A Digital Ecosystem represents a group of agents (MAS2DES-Onto:Agent) acting in a collaborative environment having a set of rules (MAS2DES-Onto:Rule) and resources (MAS2DES-Onto:Resources). It has functional time (MAS2DES-Onto:Time) and access policies (MAS2DES-Onto:AccessPolicy) about its access and usage
Complex Agent	A complex agent is an agent (MAS2DES-Onto:Agent) composed of two or more Agents (MAS2DES-Onto:Agent)
Domain	A domain refers to an area and a scope where a Digital Ecosystem (MAS2DES-Onto:DigitalEcosystem) addresses
Time	A time represents the clock of the system (MAS2DES-Onto:DigitalEcosystem) running since its launch
Rule	A rule defines conditions or constraints the agents (MAS2DES-Onto:Agent) must obey. Rules can be defined by agents (MAS2DES-Onto:Agent), which are referred as "Agent Level Rules". Then, it is also a concept of the Structural Module. There are also "System Level Rules" defined by the Digital Ecosystem. Agents (MAS2DES-Onto:Agent) complies these rules while taking any action (MAS2DES-Onto:Action). Thus, it is also a concept of the Reasoning Module
Access Policy	An access policy defines a security policy of an agent (MAS2DES-Onto:Agent) or a Digital Ecosystem (MAS2DES-Onto:DigitalEcosystem) for granting, limiting, preventing, or revoking access to a resource (MAS2DES-Onto:Resource)

1) Generic Relationships. This types of relationship exist through out the meta model and carry the same meaning in any ontology model. Commonly used relationships are: (i) *Has*, which is a relationship indicating that one entity is composed of another entity, and (ii) *IsA*, which is a subsumption relationship between entities, where one entity is a subclass (derived class) of another entity (super class).

2) Specific Relationships. relationships under this category carry unique messages and exist in different modules of MAS2DES-Onto. Table 3.7 provides the list and detailed explanation of the specific relationships among MAS2DES-Onto modules.

TABLE 3.7: Module Specific Relations

Module Specific Relations	**Description**
BelongsTo	A relationship between a Digital Ecosystem (MAS2DES-Onto:DigitalEcosystem) and a domain (MAS2DES-Onto:Domain) denoting that a Digital Ecosystem belongs to an application domain (MAS2DES-Onto:Domain)
IsGovernedBy	A relation between a Digital Ecosystem (MAS2DES-Onto:DigitalEcosystem) and time (MAS2DES-Onto:Time) which indicates that a Digital Ecosystem and its entities are subject to the same timeline (MAS2DES-Onto:Time)
Plays	A relationship between an agent (MAS2DES-Onto:Agent) and a role (MAS2DES-Onto:Role) which indicates an agent plays a specific role in the ecosystem
Receives	A relationship which shows an agent (MAS2DES-Onto:Agent) can get messages (MAS2DES-Onto:Message) using sensor (MAS2DES-Onto:Sensor) of its interface (MAS2DES-Onto:Interface)
Sends	A relationship between an agent (MAS2DES-Onto:Agent) and a message (MAS2DES-Onto:Message) denoting that an agent can transmit a message of different types using actuator (MAS2DES-Onto:Actuator) of its interface (MAS2DES-Onto:Interface)
IsRelatedTo	A relationship between a language (MAS2DES-Onto:Language) and a semantic network (MAS2DES-Onto:SemanticNetwork) to indicate that agents (MAS2DES-Onto:Agent) can communicate using the same language (MAS2DES-Onto:Language) based on a semantic network (MAS2DES-Onto:SemanticNetwork)
Modifies	A relationship between an event (MAS2DES-Onto:Event) and an action (MAS2DES-Onto:Action), a goal (MAS2DES-Onto:Goal), or a plan (MAS2DES-Onto:Plan) denoting that a captured event can enforce some changes on some of the actions, plans, or goals of an agent
Uses	A relationship between an agent (MAS2DES-Onto:Agent) and a language (MAS2DES-Onto:Language) that an agent uses language to represent its knowledge (MAS2DES-Onto:Knowledge) and messages (MAS2DES-Onto:MSG)
Executes	A relationship between a plan (MAS2DES-Onto:Plan) and a goal (MAS2DES-Onto:Goal) identifying the adequate plan to execute a specific goal
Generates	A relationship between a goal (MAS2DES-Onto:Goal) and a credit (MAS2DES-Onto:Credit) providing the credit an agent generates after executing a specific goal. It is also to indicate an action (MAS2DES-Onto:Action) can produce events (MAS2DES-Onto:Event)
ReliesOn	A relationship which indicates that a consciousness (MAS2DES-Onto:Consciousness) of an agent is derived from the cumulative effect of its knowledge (MAS2DES-Onto:Knowledge), capabilities (MAS2DES-Onto:Capability), and profile (MAS2DES-Onto:Profile)
Complies	A relationship which shows an agent must obey other agents (MAS2DES-Onto:Agent) and Digital Ecosystem rules (MAS2DES-Onto:Rule)
Captures	A relationship which shows a sensor (MAS2DES-Onto:Sensor) of an interface (MAS2DES-Onto:Interface) is also used for collecting events (MAS2DES-Onto:Event)
IsDefinedIn	A relationship between a domain (MAS2DES-Onto:Domain) of a Digital Ecosystem (MAS2DES-Onto:DigitalEcosystem) and a semantic network (MAS2DES-Onto:SemanticNetwork) to show that a semantic network bases a domain while defining concepts and their relationships
Effects	A relationship between an agent capability (MAS2DES-Onto:Capability) and its plan (MAS2DES-Onto:Plan) which shows a capability is required to act on plans
AchievedBy	A relationship which shows that a plan (MAS2DES-Onto:Plan) is realized by its actions (MAS2DES-Onto:Action)

3.5 Conclusion

Existing agent modeling languages describe agent concept in diverse ways though most do not provide a sufficient modeling capability. In this work, we present a conceptual model that can be used to model and implement MAS-based Digital Ecosystems. The key concepts in our approach were categorized in five modules (see Figure 3.7). We defined a set of concepts and their relationships in each module. These modules can be developed at various levels of detail which allow to design and develop MAS in a modularized way. The core of the MAS2DES-Onto conceptual model is an agent. MAS2DES-Onto models agents as entities with purposes (represented as agent roles, goals, and actions) and entities with internal control (represented as agent knowledge, plans, capability, and rules) to assure agents autonomy. The model also discusses about the need to consider consciousness (amongst other required behavior capabilities of an agent) to deal with events to own its reactivity character. MAS2DES-Onto also supports the modeling of agent acquaintances and interaction protocols which is significant to implement cooperative behavior of agents. MAS2DES-Onto supports the modeling of messages between agents to support communication ability. Beyond an agent, MAS2DES-Onto delivers a module to represent the Digital Ecosystem with all constituents.

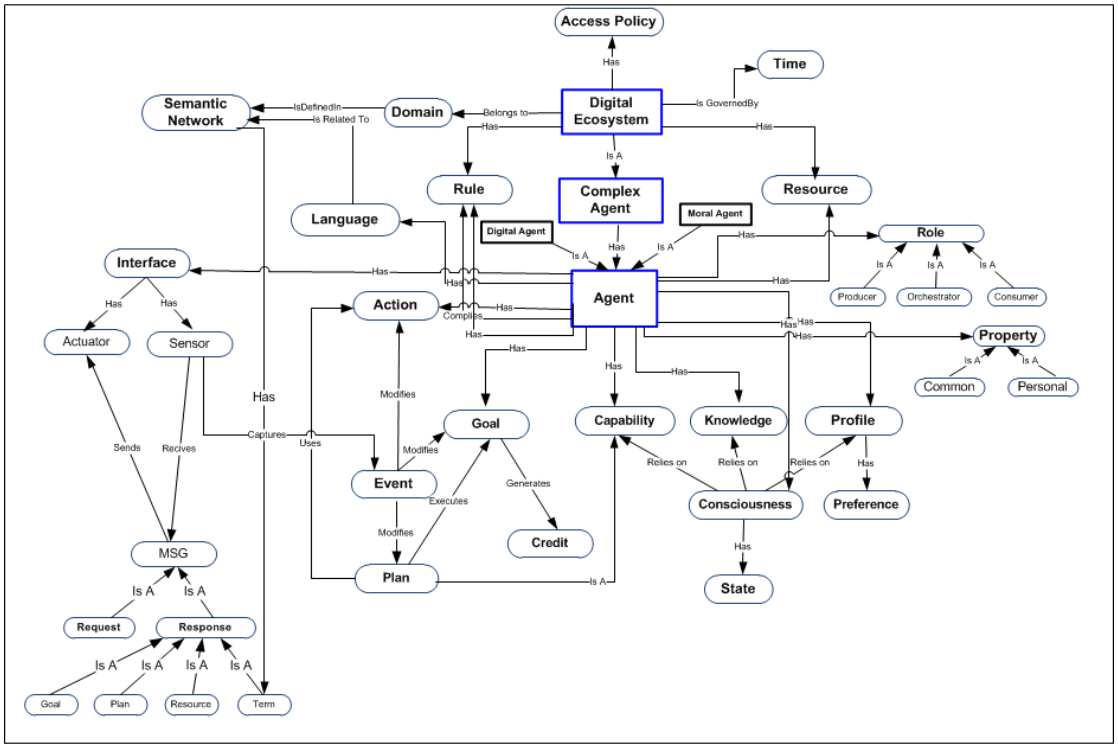

FIGURE 3.7: Detailed Conceptual Model for MAS-based Digital Ecosystem

Thus, our model captures all these requirements from MAS and Digital Ecosystem. This is done by providing a precise definition of agents in terms of its static structure, dynamic behavior, interactions with the environment, and system structure. Furthermore, agents can define their roles to play through a set of capabilities possessed that may perform upon environment objects whose effect are governed by rules and time. MAS2DES-Onto model is a key step in fully defining the design of agents to the point where a higher degree of code generation is possible. These all show the high degree of completeness and expressiveness of our model. Several concepts are specific to agent technology and they are clearly defined for using in the MAS development process. Generally, we claim that the resulting model is simple, generic, and provides a high level power of representation. To conclude, MAS2DES-Onto is modelled with the assumption of facilitating and supporting the design and development tasks of complex systems like Digital Ecosystems from ontology model.

Chapter 4

Onto2MAS: Ontology-based Framework for MAS-Based Digital Ecosystems Generation

4.1 Introduction

Since Digital Ecosystems are considered as complex systems that can be modeled on the basis of MAS, its development process becomes a very hard task and demands a considerable amount of time and advanced programming skills [106]. In order to help developers build such systems, a number of methodologies have been proposed in the literature (see Section 2.4). These methodologies vary in their approach, modelling concepts, and intended purposes. However, most of them require the usage of various tools to design the underlined MAS and to define its domain knowledge. To do so, ontologies have been commonly adopted in several studies [92, 112–115]. Most of them focus on integrating ontologies to represent the domain specification and agent knowledge to ease agents' communication and interoperability. However, none of existing modeling languages and methodologies uses ontologies to support the development of MASs, independently of the specific domain at hand [105]. To the best of our knowledge, there is no work which considers ontologies in the whole development process of a MAS-based Digital Ecosystem. Thus, we propose in this work an ontology-based MASs generation approach for Digital Ecosystems. To support this development process, we define

a conceptual model, MAS2DES-Onto presented in Chapter 3, which considers the key concepts to reach MAS and Digital Ecosystem requirements. In this chapter, we propose a complete framework called Onto2MAS (ONTOlogy TO MAS) for rapid development of MAS-based Digital Ecosystems, which integrates MAS2DES-Onto [19]. The aim of our proposed framework is threefold: (i) minimizing the efforts of MAS-based Digital Ecosystems developers by unifying tools within the same framework; (ii) separating the MASs designing and code generation tasks; and (iii) easy modification and re-usage of system models and rapid new deployments.

In the following sections, we describe the main components of the proposed framework of MAS-based Digital Ecosystems development. We implemented a first version of Onto2MAS, called OnToJade (ONtology TO JADE) [19]. The evaluation of the implemented framework shows the effectiveness of this work by demonstrating the feasibility of enabling easy and fast means to instantiate and automatically generate MAS-based Digital Ecosystems from a conceptual model into an effective implementation.

4.2 Onto2MAS Framework

Onto2MAS is a framework to support Digital Ecosystems development. Thus, the users of Onto2MAS are developers. End-users are the persons who request the specific Digital Ecosystem under a specific domain and with specific requirements. Onto2MAS provides developers the ability to: (i) design the agents in a system with their behaviors; (ii) outline the possible interactions among agents; (iii) specify the content of information exchanged among agents; and (iv) automatically generate and instantiate MAS-based Digital Ecosystems by providing an Ontology-based Integrated Development Environment (O-IDE) without requesting advanced programming skills; it enables an easy and fast means to instantiate and automatically generate MASs from the core ontology into an effective implementation. In fact, the inputs are according to end-users requirements.

We divide the whole development process of MAS-based Digital Ecosystem into three phases: *Design*, *Generation*, and *Deployment*. During the *Design process* developers define the requested concepts to meet the end-user requirements and cope with specific needs. In the *Generation process*, the concepts and rules defined in the previous process are translated into a set of agents with their related behaviors. The last process, the

Deployment process, is to create an application (mobile, web, etc.) according to the set of classes and agents previously generated. This phase deals with configuration and versioning.

4.2.1 Onto2MAS Components

Onto2MAS contains three components to support each phase of the development process: Designer, Generator, and Deployer as shown in Figure 4.1. We present these components as follows.

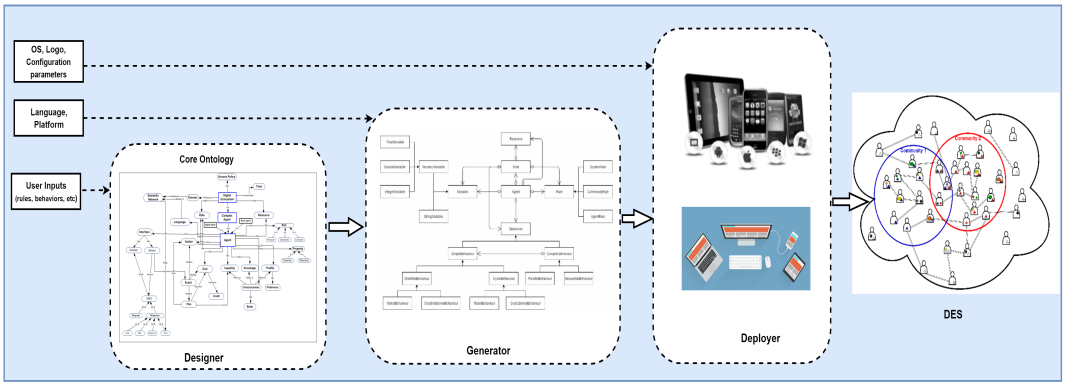

FIGURE 4.1: Onto2MAS Framework Components

4.2.1.1 Designer

This component is a means to represent the end-user/domain requirements to develop the expected MAS. To do so, the MAS2DES-Onto is integrated, with a set of agent concepts and their relationships. As shown in Figure 4.2, MAS2DES-Onto mainly serves as a base for deriving specific ontologies to assist designers in their development of different types of MASs. MAS2DES-Onto is designed with the assumption of fulfilling MAS and Digital Ecosystem requirements. As mentioned previously (see Section 3.4), MAS2DES-Onto has five modules: Structural, Species, Reasoning, Interaction, and System.

Developers only need to define and select from MAS2DES-Onto their concepts of interest from each module. Each agent concept is characterized by a set of properties such as purposes, responsibilities, roles, services to perform, information about the world it requires and maintains, and external interactions. These properties represent the end-user/domain requirements that can be adopted as they are defined by default in the

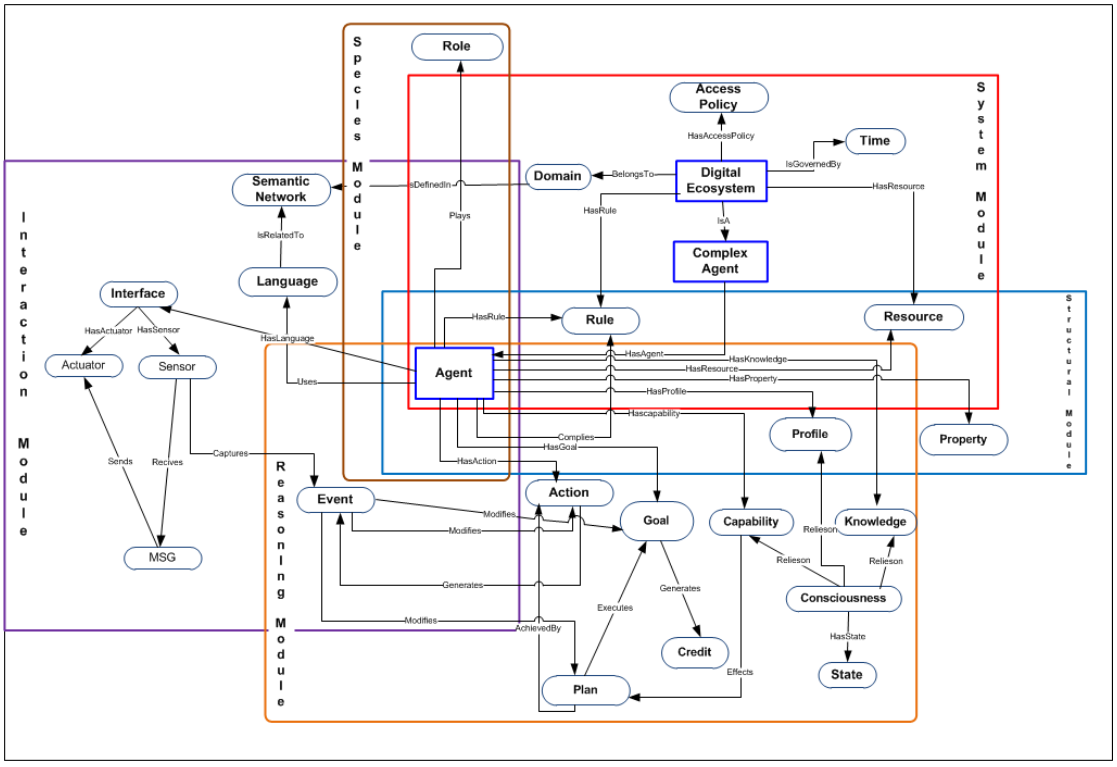

FIGURE 4.2: MAS2DES-Onto Conceptual Model

core ontology and specified/tuned using OJ language (more details about OJ will be provided in Section 4.2.2). Agents' relationships can also be defined/specified in the **Designer**. Thus, a developer would be able to generate ontology of the requested MAS-based Digital Ecosystem composed of the set of agent concepts along with related relations and requirements, as well as the number of agent instances that exist in it. This ontology will be used later by the **Generator** component of the framework.

4.2.1.2 Generator

The role of this component is to translate the concepts and rules defined by the **Designer** into agents (some are predefined and other are depending on the application at hand), and after the creation of concepts, it can be transformed into codes and agents can be generated by choosing an implementation platform. The **Generator** component takes as input the ontology generated by the developer in the previous stage. The ontology assembles pieces of concepts that realize end-users specific needs, by specifying agents hierarchy, attributes, and relations. The result of the **Generator** component (software pieces) is generated according to two other inputs: the implementation platform and the programming language.

4.2.1.3 Deployer

The **Deployer** allows to create a MAS-based Digital Ecosystem according to the set of software pieces (e.g., Java classes) and agents previously generated while considering the future system configuration. This component is responsible for two main tasks: configuration and versioning. Configuration is the way a MAS-based Digital Ecosystem is set up such as release, logo, application, while versioning is the creation and management of multiple releases of the system for various applications like web, mobile, and desktop applications. Thus, **Deployer** provides a number of ready to use applications for loading and staring up.

4.2.2 OJ Language

To complement the conceptual model of MAS-based Digital Ecosystems and enable more expressive generation of advanced systems, we designed OJ, a simplified language (or better call it "a syntax") to help the designer in the process of the specification of end-user requirements. The **Designer** can specify domain requirements with simple expressions of OJ language that are embedded to the conceptual graph to be generated and sent to the **Generator**.

OJ language is inspired from the standard SQL commands, with similar syntax and keywords. However, it also supports C/Java syntax. OJ consists of variables, basic arithmetic and mathematical operators, commands (like `SEND` and `EXECUTE`), and conditional statements. The following subsections introduce the main elements of the language.

4.2.2.1 Conditional Statement

Every programming language provides conditional statements. Likewise, OJ provides a simple `IF/ELSE` statement that can be used to execute different commands based on the results of the `IF` part of the command. This statement has two parts: condition and actions.

- *Condition* allows to test an expression. It can be written either as an SQL expression:

```
(aVariable EQUALS "aString") OR
(anotherVariable EQUALS "anotherString")
```

or as a JAVA/C++ style, such as:

```
(aVariable == "aString") || (anotherVariable == "anotherString")
```

Both styles are supported.

- *Actions* have execution elements when the *Condition* is true or not. These actions can contain commands as those detailed below.

4.2.2.2 Commands

Among the available *Commands* in OJ, `SEND` and `EXECUTE` are the main ones.

1. `SEND` is one of the most elemental operation inside a MAS and a Digital Ecosystem. It is the capability of agents to communicate between each other. Figure 4.3 summarizes the whole `SEND` syntax.

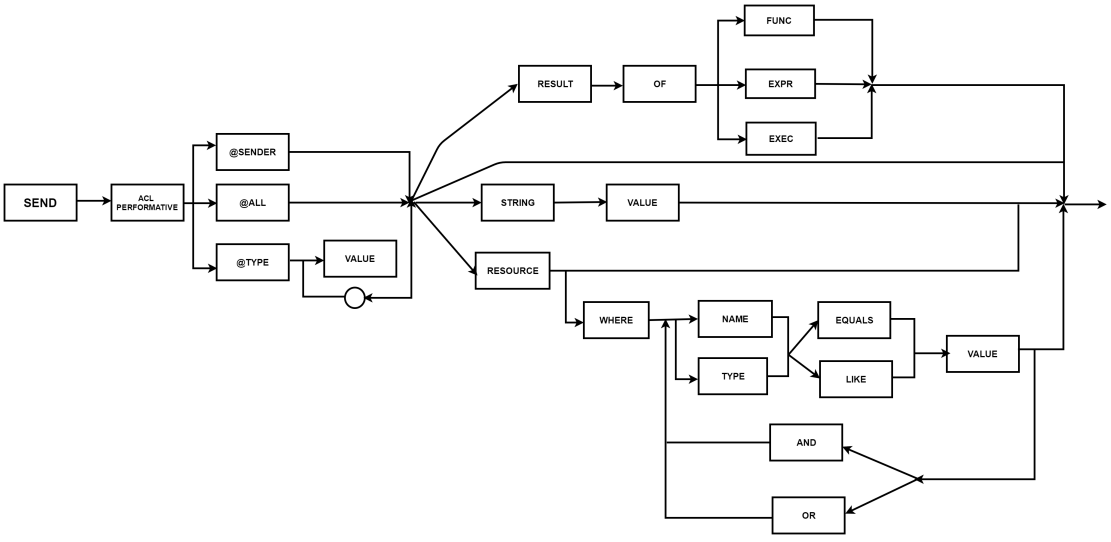

FIGURE 4.3: Send Command Syntax

This command is based on the FIPA designed Agent Communication Language (ACL) Performatives [116]. The FIPA ACL Performatives define several keywords representing different types of messages. However, the semantic behind them was deliberately left apart. Indeed, ACL Performatives are linked with different actions and are connected to a particular semantic. For instance, the `REQUEST` performative is linked to the request

action from an agent to $N > 1$ others. Our language handles all the known ACL Performatives, but no particular action will be automatically taken when receiving this type of message. The developer is free to design what (s)he wants, from the very beginning to the very end. The `SEND` command is separated into three main sections: `how`, `who`, and `what`.

- `How`: states in what way a message will be sent. As we previously said, `SEND` uses FIPA ACL Performatives to handle message communications and provides the possibility to create a message using one of the 21 available types of messages (`REQUEST`, `CONFIRM`, `INFORM` ...). For example, if Agent X wants to send a request to Agent Y, then the action of Agent X can be written in OJ as: *`"SEND REQUEST @Agent Y"`*. In this example, "SEND" represents the command type, and "REQUEST" indicates the actual message to communicate.

- `Who`: defines who will receive the message. `SEND` can send the specified message into four different ways:

  1. The developer may want to send a message in response to a previously received message . Specifying the option `@SENDER`, `SEND` will automatically create a reply to this message.

  2. The message may be conceived to be received by everyone on the platform (e.g., a broadcast). Specifying the option `@ALL`, the message will be created and received by everyone on the platform. For example, the command *`"SEND AGREE @ALL STRING "hello""`*, sends a message of type "AGREE" to all the agents of the system and containing the chain of character "hello".

  3. The message may be intended to be received by a particular group of agents (e.g., a multicast). These agents, which have been previously registered using a specific service name, and can be contacted using a specific `tag`.

  4. The designer can choose to send the message to a particular agent (e.g., a unicast). Using one of the agents name inside the ontology, `SEND` will automatically contact this particular agent.

- `What`: defines the actual message to be communicated. `SEND` allows the developer to fill in the message content slot. The developer can choose to create a message or leave the content slot empty. This can be useful for multiple reasons. If the

receiving agent was designed to be triggered by the Performative alone, the content slot is then meaningless. However, the content slot can contain different types of information such as a string or a link to a file.

2. `EXEC` is designed to make an agent executes codes that come from other agents. An agent has the possibility to execute files, links, and other resources which are received from other agents. Results of this execution can be send to the sender if needed. Executions are done using parameters specified by the developer. Agents will automatically run the specified file with the right parameters. Figure 4.4 shows graphically the syntax of this command.

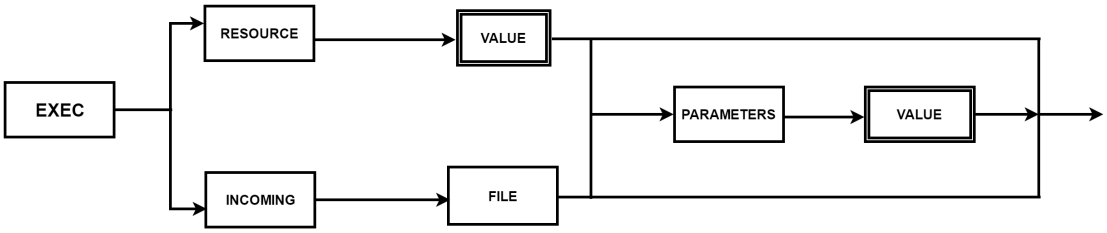

FIGURE 4.4: Execution Command Syntax

4.2.2.3 Basic Arithmetical Operations

Like all other programming languages, OJ language provides statements to operate over user defined variables. Arithmetic operators perform mathematical operations on two expressions of one or more of the data types of the numeric data type category. Five basic operations are supported: addition, subtraction, multiplication, division, and modulo. The syntax for the different operations:

```
(ADD|SUB|MUL|DIV|MOD) var EXPR
```

where `var` is a user (i.e., developer) defined variable and `EXPR` is a mathematical expression, such those presented in the next section.

4.2.2.4 Mathematical Expressions

Every agent instantiated inside the system will be provided with an Arithmetical Logical Unit (ALU) which can calculate mathematical expressions, even complex ones. These

expressions are standard mathematical expressions except that they can be constructed using different values. OJ provides operators that combine results from two or more expressions into a single result set. For example, to obtain a random integer in the range $8 \leq r < 12$ using OJ language, we could use the following mathematical expression:

```
(8 + rand(4))
```

Mathematical expressions are constructed using values and operands. Values are of three different types:

- *User Inputs.* These are the most simple values that can be used inside an OJ mathematical expression. The designer/developer can directly add constants to the expression. An example of a constant is `4` in the previous example.
- *User Variables.* These variables can be initiated by the developer but most importantly can also be used as a value to construct a mathematical expression. The name of the variable will be replaced by its value when it comes to mathematical operations.
- *Functions.* Functions are built-in mathematical functions that can be called inside every appropriate commands. For the time being, we only considered two functions: `rand` and `sum`. The previous example shows how such functions can be called, with an example of the function `rand` which will return a random number between 0 and the number passed as a parameter. `sum` will however return the sum of all the numbers passed as parameters to the function. Parameters can be constants provided by the user, the value of a user variable, or even the result of another function.

  The following example shows how functions can accept all the three possible types of parameters.

  ```
  (sum(rand(10), rand(15), variable, 4))
  ```

4.3 OnToJade: Implementation of Onto2MAS

To prove the feasibility of our framework, we develop a prototype, called OnToJade (ONtology TO JADE) by using JADE[1] as the implementation platform, Protégé OWL editor, OJ Language, and JAVA programming language. We made experimental tests to show its performance and workability. In this section, we describe the components of the prototype as show in Figure 4.5. In OnToJade, we only consider **Designer and Generator** components.

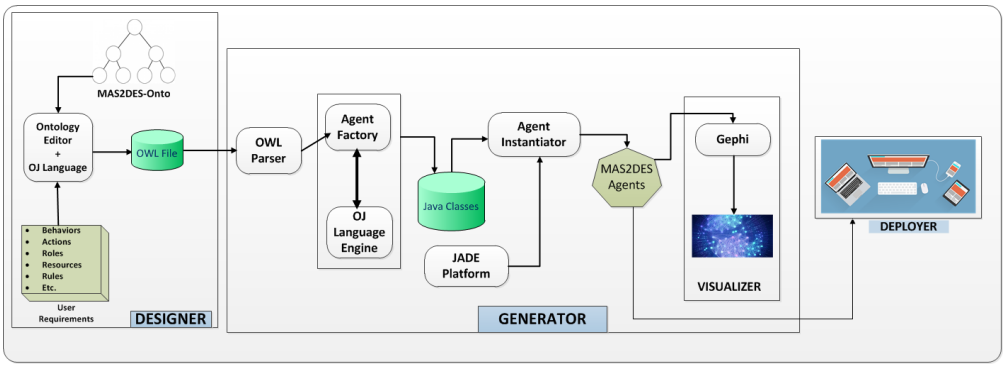

FIGURE 4.5: Architecture of OnToJade Prototype

4.3.1 Designer Component of OnToJade

The main advantage of OnToJade, as a prototype of our framework, is to provide to the developer a tool to easily create a MAS-based Digital Ecosystem with the only concern of the design of it. Thus, the developer uses the **Designer** component to create an ontology. To do so, the developer considers end-user requirements to derive necessary concepts from MAS2DES-Onto. Based on the derived ontology, developer specifies the right definition and functionalities of *Agents* for a particular system and their *Behaviors*, *Roles*, *Rules*, and *Resources* using Ontology Editor and OJ Language, which are represented in an `owl file`. The specifications as ontology model (`owl file`) are stored in a repository to serves as input to the *Generation process*. In the next section, we provide details of the derived ontology.

[1]http://jade.tilab.com

4.3.1.1 Derived Ontology

The first step of a developer is to create an ontology that represent the intended MAS-based Digital Ecosystem.

This ontology is mainly derived from the MAS2DES-Onto provided in the previous chapter (see Figure 4.6 (A)), which intends to assist the developer in defining its MAS-based Digital Ecosystem. Thus, as shown in Figure 4.6 (B) we define a simple ontology from MAS2DES-Onto. The following concepts are taken from MAS2DES-Onto modules: Digital Ecosystem, Agent, Resources, Role, Rule, and Action. These concepts are the basics that any agent in Digital Ecosystems should owned. Thus, we are limited to these concepts for the purpose of OnToJade prototype. Next, we detail the concepts, individuals, and their relationships as follows.

A) Main concepts

A Digital Ecosystem is represented as a group of interacting and collaborating agents. An *agent* is the main concept in the proposed core ontology representing any individual in the Digital Ecosystem. Each *agent* is mainly defined through its *Behavior* (or *Action*) and *Role* within the Digital Ecosystem. It must follow a set of *Rules* and can share a set of *Resources* in the Digital Ecosystem. In the following, we detail each of the linked concepts.

1. **Behavior**: as this concept is equivalence of *Action* in MAS2DES-Onto, it represents the actions executed by an *Agent*. An *Agent* can have several behaviors, which can be: *Simple Behavior* or *Complex Behavior*.

    - *Simple Behavior* is designed to be the most simplistic action to be executed by an *Agent*. A *SimpleBehavior* can be:

        – *OneShot Behavior*. It allows executing actions only once. *OneShot Behavior* is the best choice to perform actions such as registering an agent, triggering the system, etc. This behavior can be executed when an agent is instantiated and only once. Other simple behaviors inherit from this concept, such as: (i) *Waker Behavior*, which is defined to perform an action after a certain delay; this behaviour is used for multiple purposes such as doing an action after the whole system is initiated; and (ii)

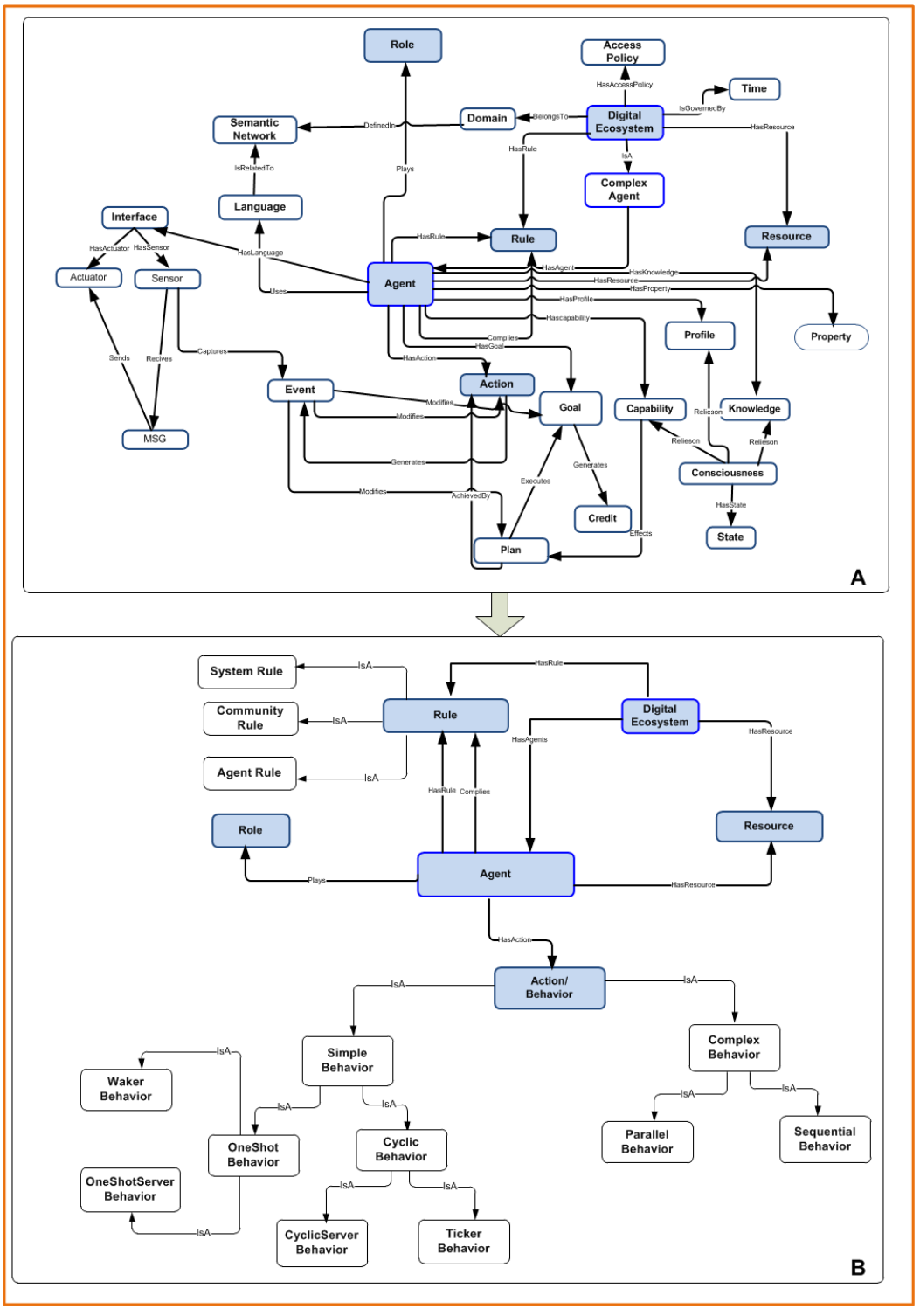

FIGURE 4.6: Derived Ontology for OnToJade

OneShotServer Behavior, which inherits from *OneShot behavior* except that an action is executed by different agents after receiving a particular type of message.

– *Cyclic Behavior* executes its actions infinitely and will not be stopped

until the agent dies or cannot pursue its task(s). It is extended through: (i) *Ticker Behavior*, which executes itself only after a specified delay before starting over; and (ii) *CyclicServer Behavior*, which is the cyclic equivalent to *OneShotServer Behavior* and executes its actions everytime a message matches the condition; this behavior is useful to create agents that will respond to different types of requests (such as content requests, support requests).

- *ComplexBehavior*. It embeds two or more simple behaviors which can be organized through two types: (i) *Sequential Behavior*, which consists of a set of behaviors that are executed one after the other; it terminates when the last behavior ends; and (ii) *Parallel Behavior*, which includes a set of behaviors that are executed in parallel; it terminates when all behaviors are finished.

2. **Rule**: this concept is primarily to address a Digital Ecosystem. A Digital Ecosystem needs to provide some rules to be followed by the agents of the system. In Digital Ecosystems, agents can organize themselves, for example according to special interests, in *communities* where they also can follow additional rules. Thus, rules can be designed to be followed by all agents, particular agents, or by a community of agents in the Digital Ecosystem. Accordingly, three different types of rules are set: (i) *System rules* are applied to all agents and are important for the integrity of the system (for example, a Digital Ecosystem must , at least, have two agents); (ii) *Community Rules* are applied to some agents (for instance, a community can have a rule which states an agent should give a priority to a request that comes from an agent from its community than other); and (iii) *Particular Agent* rules concern only one agent in the Digital Ecosystem (for example, an agent A can have a rule not to communicate with agent B because of its bad reputation). Thus, those three different types of rules are hierarchically ordered which means that when a rule has a higher level, it has priority over a lower one.

3. **Role**: this concept is designed to create agent roles in the Digital Ecosystem. The created roles have multiple features, such as behaviors, rules, and variables.

4. **Resource**: this concept allows to represent a given resource (file, folder, application, etc.) to be shared among several agents.

5. **Variable** is used to define actions within behaviors. Variables can be created using the basic types: *Integer, Double, Float* for *Numeric Variables*, and *String* for *String Variables*.

B) Individuals

Individuals are instances of the defined concepts. They represent the different objects to be instantiated in the final generated MAS-based Digital Ecosystem. Each individual has several *Properties* representing its main features. Some properties are listed below:

- *action* is the main property for behaviors and rules.
- *reactsTo* is used for special behaviors. For example, it is used to trigger the specified behavior when a particular message is received by the linked agent.
- *timer* is used to set the delay in Waker Behavior and Ticker Behavior concepts.
- *defaultValue* is used for variables' instances. After the creation of the variable inside the system, this property will set its value to the specified one.
- *instances* is a special property that can be used to create $X > 1$ instances of an agent concept.
- *path* is used to determine the path of a specified resource location.

C) Relations

Relations are used to represent semantic relations (links) between agents and corresponding individuals. Some are listed here:

- *hasBehavior* is a relation between an instance of a behavior and either an agent or a role.
- *hasRule* is a relation between an instance of a rule and either an agent or a role.
- *hasVariable* is a relation between an instance of a variable and either an agent or a role.
- *hasResource* is a relation between an instance of a resource and either an agent or a role.
- *hasRole* is a relation between an instance of a role and an agent.

4.3.1.2 Creating Ontology Representation Files

Based on the derived ontology, the next step is to create an ontology file using Protégé Ontology tool, from which we can define agents, actions, and other concepts. Figure 4.7 shows the graphical interface of Protégé OWL tool[2]. It is a very popular open source ontology development tool that was developed at Stanford University. This tool is selected as it is the best-known ontology editor that supports: loading and saving OWL and RDF ontologies, editing and visualizing OWL classes and their properties, defining logical class characteristics as OWL expressions, editing OWL individuals, and executing reasoners [117]. Protégé also enables ontology designers can instantly create individuals of their ontology. It is implemented in Java; and it is becoming the de-facto standard OWL editor[3]. Hence, the derived ontology is designed using this tool.

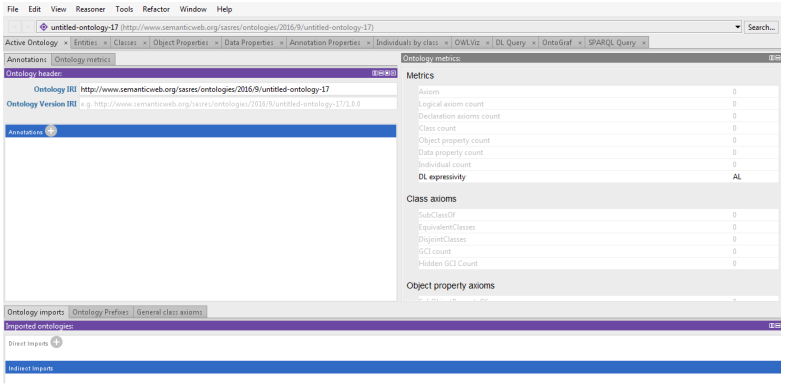

FIGURE 4.7: Protégé Ontology Tool Interface

1) Classes

In Figure 4.8, a screen capture of the ontology editor is presented showing classes which have been defined according to our derived ontology model. The Agent class is the super class of all the other derived agent classes. In addition to Agent class, there are five important classes for the definition and implementation of a MAS-based Digital Ecosystem: Behavior, Resource, Role, Rule, and Variable. When it is necessary, instances of these classes will be created.

Behavior class is the super class of the different possible behaviors which the agents can execute when they launch inside the system. Different types of simple behaviors are defined which inherit the class Behavior. Of course, when the need comes, the developer

[2]http://protege.stanford.edu/plugins/owl/index.html.
[3]http://protege.stanford.edu/plugins/owl/index.html

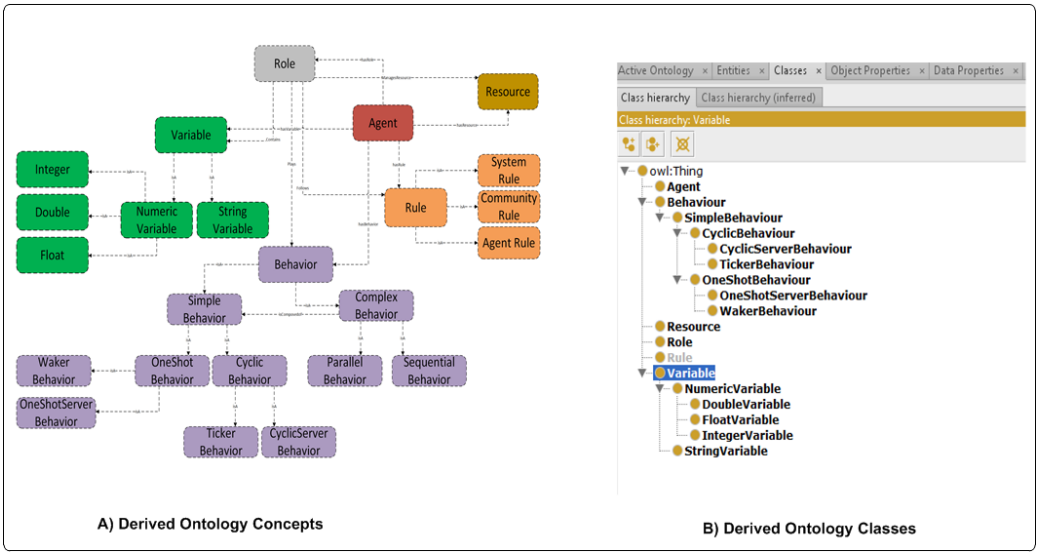

FIGURE 4.8: Derived Ontology Classes

can add more and more behaviors depending on the complexity of the designed Digital Ecosystem. The Concept Variable is the super class of all the data types which cannot be integrated inside the previous classes. Resource is used to define data and executable files an agent can have. Rule class is also considered to define rules that the agent must obey. Depending on the requirements of the end-users, the developer could define different types of rules at various level such as system, community, or agent. However, these rules must be designed in a hierarchy to show that system level rules are more important than community rules and community rules have priority than agent rules. This hierarchy is necessary to resolve when there is a conflict between two rules from different levels.

2) Top Data Property

Top Data Property represents the various attributes that the authorities of the classes can possess. Figure 4.9 displays the various types of top data properties which are integrated into our ontology. For instance, *behaviourProperties* is the super-property which has three sub-properties: *action*, *reactsTo*, and *timer*. These properties tell that an agent can take action initiated by itself (like registration, request, inform) or an agent can react for requests that come from another agent in a defined time duration, as presented in Figure 4.10.

3) Individuals

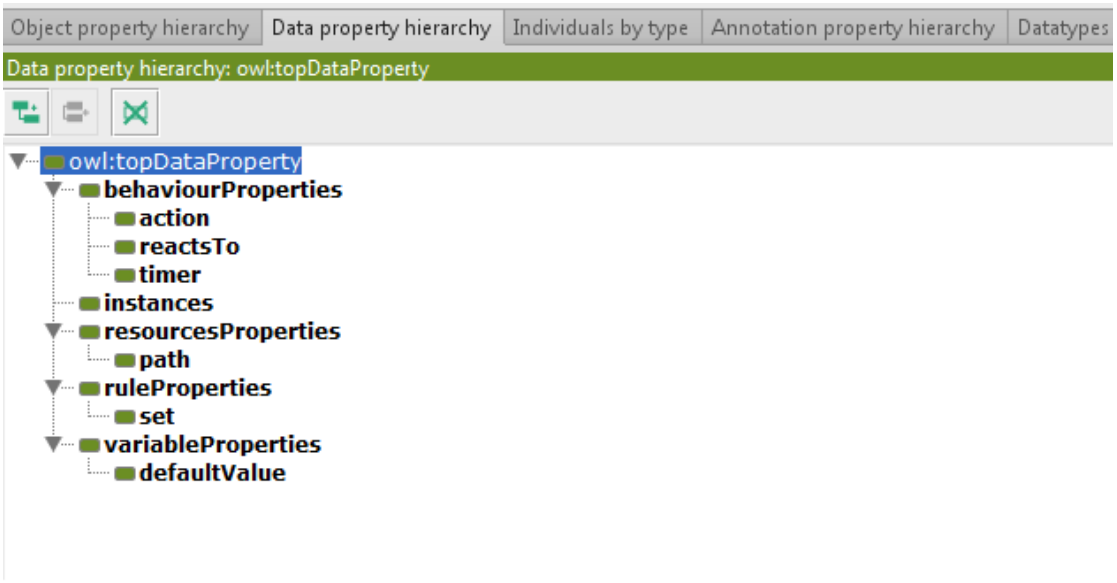

FIGURE 4.9: Derived Ontology Top Data Property

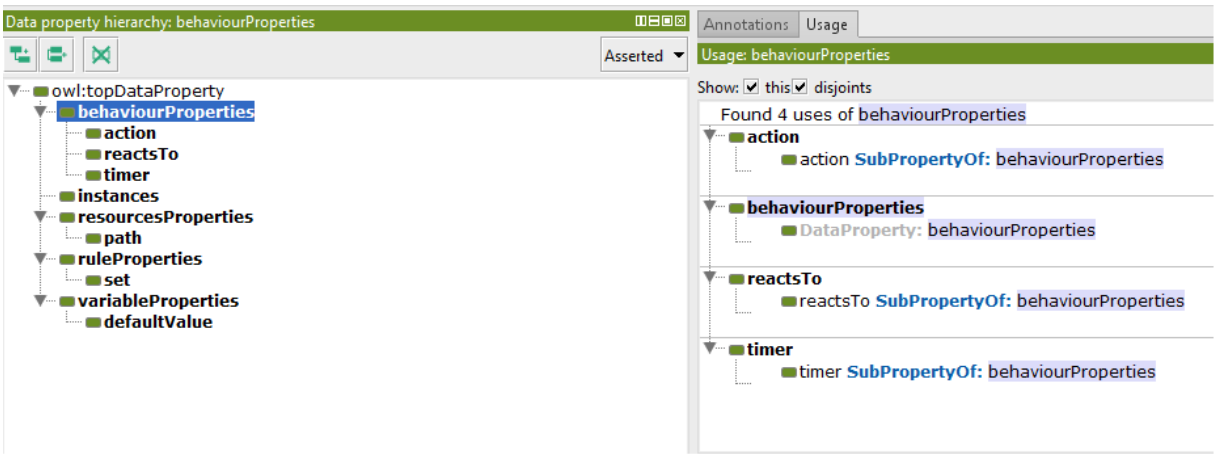

FIGURE 4.10: Behavior Property Sub-properties Description

Individuals represent instances of the classes in the ontology. The number of instances in each class could be determined by the developer taking into consideration the scope and objective of the intended MAS-based Digital Ecosystem. As we can see in Figure 4.11, four agents are created under the super class Agent and two types of roles (*ProcessingRole* and *ProviderRole*) that contain three resources.

As well, each individual can be described more in association to other concepts. For instance, Figure 4.12 shows that **AgentA** is a type of agent which has a *Requestwaker* behavior and a *ResourcesFolder*.

In order to link one class with another class, the ontology editor supports the developer

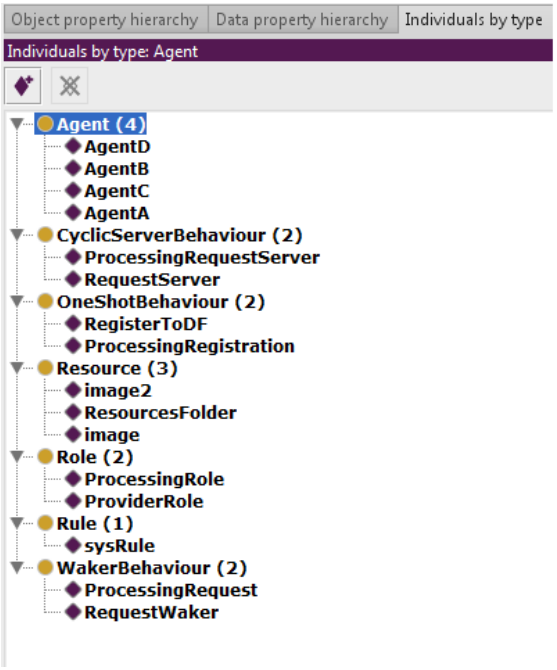

FIGURE 4.11: OnToJade Individuals by Type

to define object properties to show the relationship between two classes. In Figure 4.13, the object property *hasBehavior* indicates that **AgentA** owns requesting contents and processing actions which imply this agent is playing a requester role. Thus, from the behavior of an agent, we can deduce their role and resource.

The developer can detail more its requirements with the support of the OJ language. From Figure 4.14, we can notice that a behavior called *ProcessingRequestServer* is further described using the OJ language as an action which *"reactsTo Request"* and *"SEND INFORM @sender Result of EXECUTION"*.

To summarize, the final out as a MAS-based Digital Ecosystem is a reflection of the inputs encoded by the developer. Hence, the developer should clearly understand the end-users requirement to well address their need. Accordingly, the necessary classes, instances, and relationships must be created and stored for the next step of the development task. In fact, it is always open that the developer might make modification to the model when necessary.

4.3.2 Generator Component of OnToJade

Once the developer does all the required tasks, the next step is up-to the **Generator** component to generate all the necessary agents and instantiate the system.

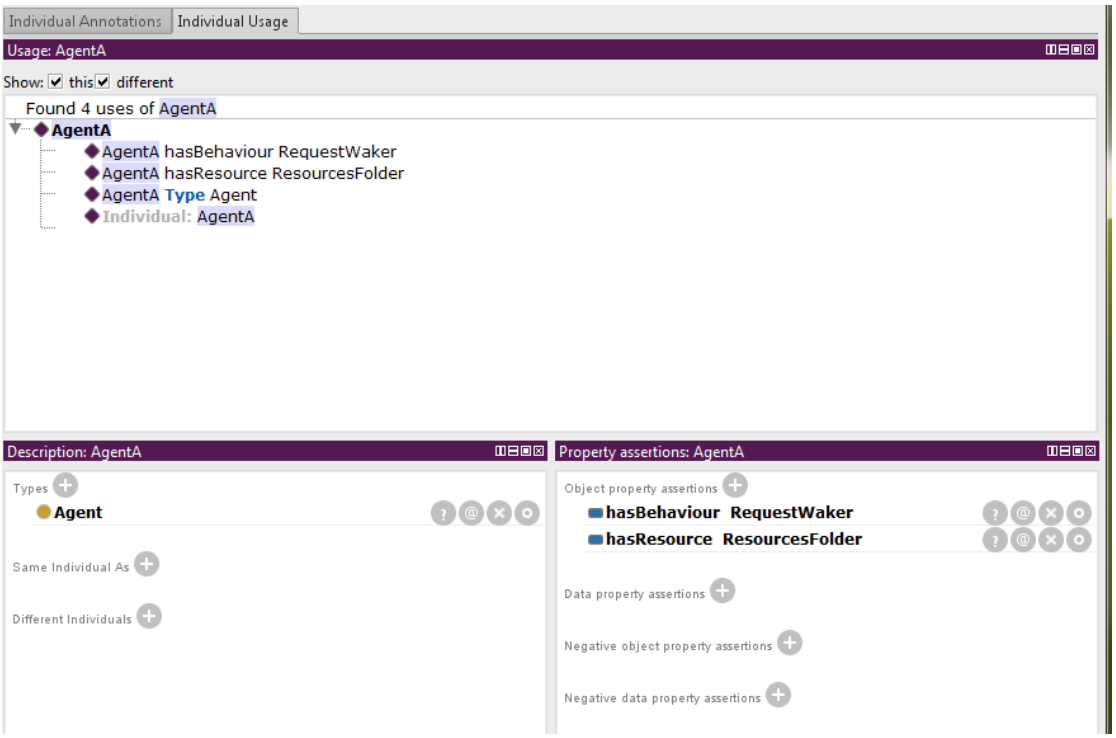

FIGURE 4.12: OnToJade Individual Agent Description

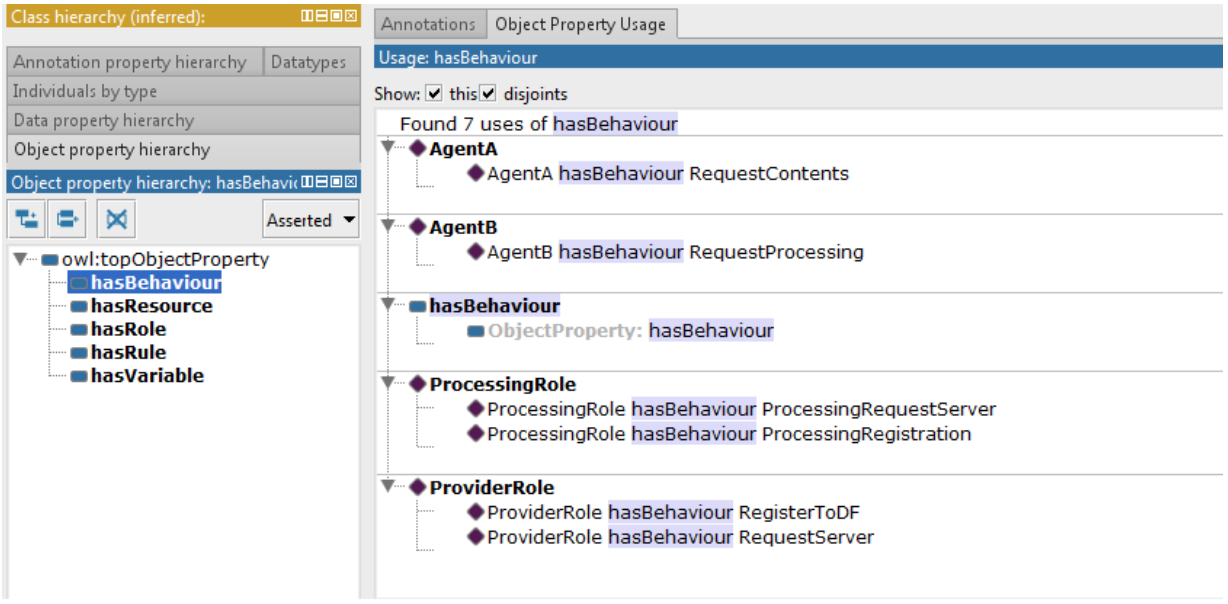

FIGURE 4.13: OnToJade Agent Property Description

To accomplish this, the Generator uses as input the ontology file (i.e, `owl file`) provided by the **Designer** component to convert it into classes of agents. This components runs different tasks to deliver the expected from accessing ontology files to visualizing the generated system. As a consequence, it has sub components as shown in Figure 4.15 and described as follows.

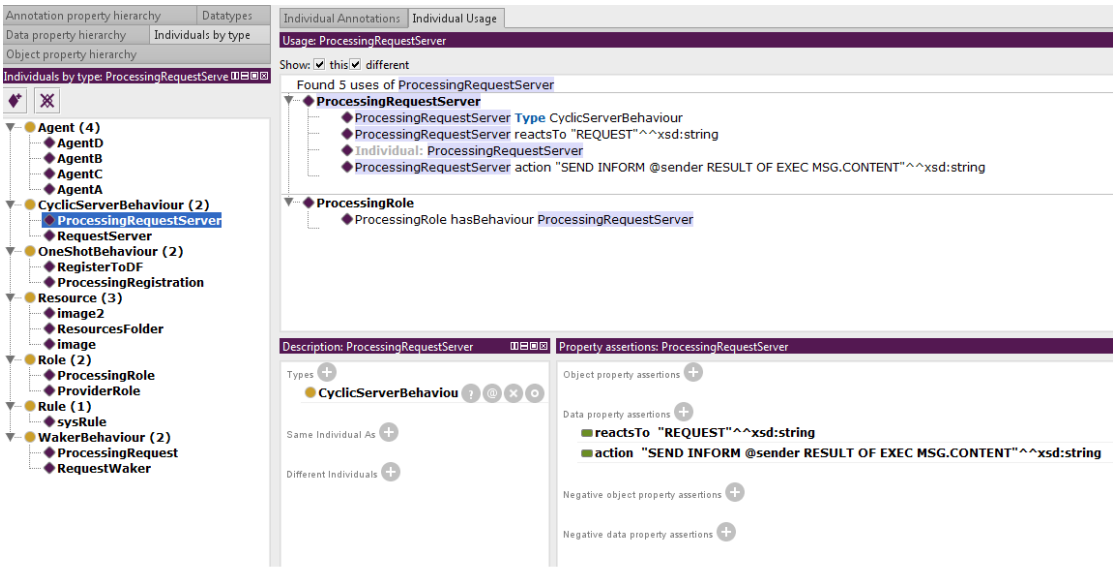

FIGURE 4.14: Behavior Description using OJ Language

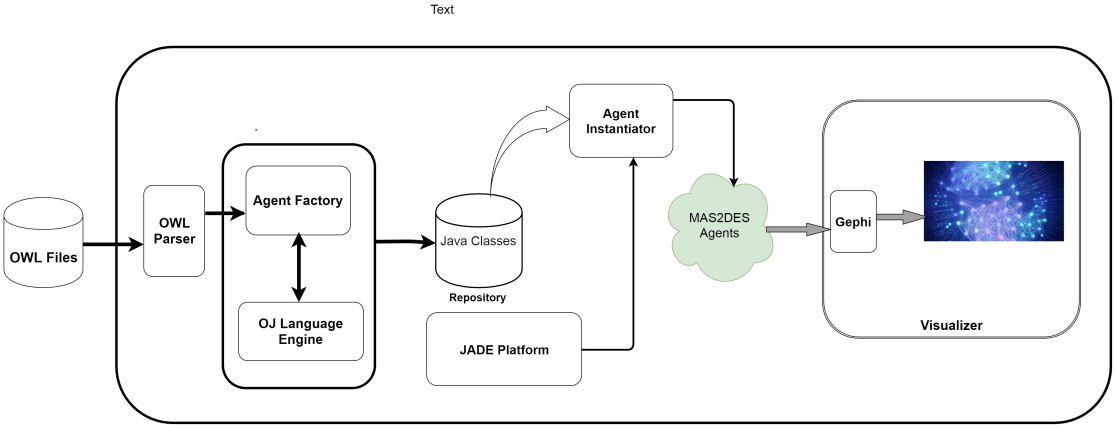

FIGURE 4.15: Generator Component of OnToJade

4.3.2.1 OWL Parser

The first step of the *Generation process* is to retrieve the inputs of the developer provided in a form of `owl file`. In OnToJade, the basic idea is to create a set of Java classes from an OWL ontology such that an instance of a Java class represents an instance of a single class of the ontology with its properties, class relationships, and restriction-definition. Thus, OWL Parser component employs a library called Jena[4] to smoothly access the OWL files. Jena is an open source programming toolkit, using the Java programming language for building Semantic Web applications[5]. It provides a extensive Java libraries for helping developers write codes that handles OWL, RDF, and SPARQL[6]. OWL Parser

[4]http://jena.sourceforge.net
[5]http://jena.apache.org/
[6]https://www.w3.org/2001/sw/wiki/Apache_ Jena

with the support of this library is used to retrieve OWL files to the OnToJade environment for proper mapping and creation of necessary agent instances and properties

OWL Parser reads every information to create Java objects representing them. Each concept found in the ontology has its counterpart inside the Parser. First, OWL Parser collects every information about the *Behaviours, Rules*, and *Variables* defined in the ontology. These concepts are analyzed to find their *Attributes*. Then, the Parser finds *Roles* defined inside the ontology. It also creates objects representation of these *Roles* following several *Rules* as the different *Relations*. The parser follows these object *Relations* and constructs the *Role* by linking it to the previously created *Behaviour, Variables,* and *Rules*. Finally, it finds *Agents* in the ontology with their *Attributes* and *Relations* to link them with their *Behaviour, Variables, Rules*, and *Roles*.

4.3.2.2 Agent Factory

The next step of the *Generation process* is code generation for creating *Agent* classes. The OWL Parser module automatically calls the Agent Factory module to build the different *Agents* found in the ontology. *Agents* are the program objects that describe them but are only basic data and OJ instructions. In order to build instances of Java classes, we use the standard Javassist[7]. It is a class library for editing bytecodes in Java; it enables Java programs to define a new class at runtime and to modify a class file when the JVM loads it. It is also important to access the values of the properties of the class and to maintain class relationships present in the ontology. The detailed tasks of this module are given below:

- **Behaviour creation**. At this point, the *Behaviours* found inside the ontology are only represented by an Object which describes the desired class of the *Behaviour*, the action designed to be fit in this *Behaviour*, and particular information such as the timer for the *WakerBehaviour* or the *TimerBehaviour*. These OJ instructions will be translated to understandable JAVA code using `javassist` implemented in the Language Engine module, described below.

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

- **Rules and Variables creation**. *Rules* and *Variables* are also represented as JAVA Objects containing their information. The Agent Factory creates new Objects based on them so they can be used by the linked *Agent*.

- **Resources creation**. *Resources* are handled by an object created automatically and found in a folder path. Every file inside this folder (including potential subfolders) will be stored in a data structure. This structure will be associated to the *Agent* which is important to access every files of the folder in its *behaviors*, *messages*, etc.

To facilitate the creation of *Agents*, OnToJade provides an *Agent*, called `ExtendedAgent class`, to be the super class of each *Agent* of the system (except the Coordinator Agents, which have their own classes) that will be instantiated. This *Agent*, based on JADE super-class Agent, offers multiple new features:

- **Automatic subscription**. `ExtendedAgent` will automatically send a message to the Moderator Agent to subscribe to it. After the *Agent* setup is over, it will send messages with information required by the Moderator to be able to run on the platform.

- **Extended send method (sendMsg)**. It provides a method to send a message to *Agents*. This method is the starting point for every interaction in a MAS. `ExtendedAgent` provides the method `sendMsg` which has an additional task: every time the *Agent* sends a message, it will also send a message to the Moderator Agent to let it know that an exchange has been made. This is important to reset the `timeToQuit` time of the *Agent*.

- **Define global parameters**. `ExtendedAgent` is created as standard Agent on the JADE platform, with a name and parameters. However, an instance of this class (i.e., an *Agent* of our system) will automatically parse those parameters and construct itself from them. The system will give this *Agent* as parameters everything that can be linked to this *Agent (Variable, Rule, Behaviour*, etc.). The *Agent* will do actions depending on the nature of the current parameter automatically. For instance, if the *Agent* receives a *Behaviour* class, it will create an instance of this *Behaviour* and attach itself to it.

4.3.2.3 OJ Language Engine

The Language Engine module is the core of this component. Its task is to translate the different OJ statements into a Java equivalent ones. Moreover, to help the user, the Engine analyzes the OJ statements syntax, tries to resolve the problems and correct errors, and reports errors in order to the developer can correct and resubmit the right requirements. If everything is right, this module interacts with the Agent Factory module, to finally create the proper Java classes of all *Individuals* in the MAS.

4.3.2.4 Agent Instantiator

This component creates the corresponding system based on ontology models and Java classes obtained from the previous components with the support of JADE platform. JADE has been commonly adopted by MAS community which is distributed by Telecom Italia[8]. JADE system supports coordination between several agents and provides a standard implementation of the communication language FIPA-ACL[9], which facilitates the communication between agents. Thus, we constrained ourselves to use JADE for running OnToJade as it is one of the well-known and frequently used open source agent platforms.

At this stage of the *Generation process*, the developer initial models are converted into the intended system. To realize this, different kinds of agents are created and then the system is instantiated. To make functional the system, this component considers three **Coordinator Agents**: Agent Management System (AMS)and Directory Facilitator (DF) and the Moderator Agent.

AMS and DF JADE agents are automatically activated at start-up of JADE. The AMS is the agent who controls the platform over access and use of the JADE platform. It is the agent who can create and destroy other agents, destroy containers, and stop the platform. The DF is the agent who provides a directory which announces which agents are available on the platform. Based on these two special agents, we implemented an additional special agent, called **Moderator Agent**, to coordinate the Digital Ecosystem and easily get information from the system (especially for the representation of the system). It is incorporated to supplement

[8] http://jade.tilab.com
[9] http://www.fipa.org/

the platform. Thus, OnToJade automatically instantiates these three coordinator agents with every MAS created. Since AMS and DF are common JADE agents, we only describe below our **Moderator Agent**, which is in charge of implementing system level *Rules* and complies the following functions.

– **Handling idle Agents**. An *Agent* which is idle inside the Digital Ecosystem is not interesting for the system. Every *Agent* in the system is configured to respond to the ACL Performative `UNKNOWN` message (sent by the Moderator as a `ping`) with a `NOT_UNDERSTOOD` one (which means `I'm alive`). If the Moderator sends the message and does not get any response from an *Agent*, this means it is in an idle state. The Moderator will first send an alert message to try to trigger this *Agent*. If the *Agent* still does not respond, the Moderator will ask the AMS Agent to kill it. Algorithm 1 clearly shows this task.

Algorithm 1 Handling idle Agents algorithm

```
while true do
    while alert list is not empty do
        Send alert message
        if an Agent is not responding then
            Ask AMS to kill the agent
            Remove agent from the alert list
        end if
    end while
    for all Agents do
        Send UNKNOWN message
        if an agent is not responding then
            Add it to the alert list
        end if
    end for
end while
```

– **Resources declaration**. *Agents* in the Digital Ecosystems will have to declare their *Resources* to the Moderator. The developer can modify a value using the system *Rules* representing the minimum number of *Resources* that an *Agent* must fulfill the requirements to join the Digital Ecosystem. Every *Agent* will send a `SUBSCRIBE` message to the Moderator Agent as soon as it can. The Moderator will catch these messages and fill its list of *Agents* contents. Algorithm 2 presents this process. Another part of the *Resources* handling is that the Moderator will refuse the entrance to *Agents* that do not

meet the requirements. That is done by comparing the amount of declared resources to the defined threshold (specified in a default values file).

Algorithm 2 Handling resources declaration algorithm

```
while true do Receive SUBSCRIBE message
  if sender is not known then
    Add it to the Agents list
  else
    Add sent content to this Agent's
    Resources list
  end if
end while
```

– **Managing uncooperative Agents**. Every *Agent* has a modified version of the `jade.core.agent send` method. This modified version will send another message to the Moderator to let it know that an exchange of information has been done. Moreover, the Moderator keeps a trace of *Agents* exchange. Every *Agent*, after being registered at the Moderator Agent, will have a counter (called `timeToQuit`) which will decrease over time. The Moderator will reset `timeToQuit` to the default value (which can be modified inside the ontology) every time it receives a trace of an exchange for an *Agent*. The Moderator will also decrease the `timeToQuit` values for every *Agent* (except AMS, DF, and itself) every $X > 0$ milliseconds, where X can also be modified inside the ontology. If an *Agent* `timeToQuit` reaches the zero value, its name will be inscribed in the `alert list` of the Algorithm 1. This process is displayed in Algorithms 3 and 4.

Algorithm 3 Handling uncooperative Agents algorithm – Part 1

```
while true do
  Receive message(s)
  Reset timeToQuit for the sender AND the receivers
end while
```

Algorithm 4 Handling uncooperative Agents algorithm – Part 2

```
while true do
  Wait(X)
  Decrease timeToQuit values
  if timeToQuit==0 then
    Add it to the alert list
  else
    Break;
  end if
end while
```

4.3.2.5 Visualizer

This component allows to graphically represent the instantiated system. OnToJade uses the Gephi toolkit[10] to make visualize, as a graph, the generated system to end-users. Gephi provides tools to create different kinds of graphs, however the user interface (UI) uses the oriented graph toolkit. *Agents* are the nodes in the graph. Edges represent the exchanges among *Agents*. The Moderator informs Gephi each time when something happens inside the system. For instance, the representation will update itself every time a new message is sent or when an *Agent* dies. Edges will be automatically deleted after some time to avoid a lose of visibility of the system when a huge number of messages are sent at the same time (it deletes the oldest first).

4.4 Experimental Tests and Results

The experimental tests are designed to confirm the easy-to-use and quick development support of the proposed framework. Three different kinds of tests are conducted. The first test is to show the performance of the proposed framework. The second test focuses on agent interaction in the system; and the third test is to check the function of Moderator Agent in enforcing system level rules. All these tests were executed in an Intel(R) Core(TM) i7-2600 CPU, 64 bits, 3.4GHz, 8GB of RAM memory, with Windows 7.

4.4.1 Performance Test

The objective of this test is to prove that the proposed framework supports developers better than existing approach in generating any MAS-based Digital Ecosystem. To show this, we compare the process of automatically generating MAS-based Digital Ecosystems using OnToJade against the usual manual way by using platforms such as JADE and Java. To proceed with this test, three Digital Ecosystems (hereinafter referred as DES) are created: (i) **DES 1**, which contains the minimum possible number of agents (two agents); (ii) **DES 2**, with ten agents; and (iii) **DES 3**, which has fifteen agents. Three measures are calculated: the amount of les managed by developers, the number of code lines, and the time required to create the Java les and to instantiate the system. Each

[10]https://gephi.org/toolkit/

DES in OnToJade was generated 500 times and the average measure is taken as the nal result.

TABLE 4.1: Measures to Generate DESs

DESs	Measures	*OnToJade*	*JADE/Java*
DES 1	Number of managed files	1	4
	Number of code lines	∼10	∼50
	Time to create file(s)	20 sec.	2 min.
	Time to instantiate	1.1184 sec.	0.621 sec.
	Total Time	**<1 min.**	**<3 min.**
DES 2	Number of managed files	1	6
	Number of code lines	∼15	∼70
	Time to create file(s)	30 sec.	3 min.
	Time to instantiate	1.197 sec.	0.629 sec.
	Total Time	**<1 min.**	**<4 min.**
DES 3	Number of managed files	1	8
	Number of code lines	∼20	∼90
	Time to create file(s)	45 sec.	4 min.
	Time to instantiate	1.214 sec.	0.648 sec.
	Total Time	**<1 min.**	**<5 min.**

Tables 4.1 presents the results, in which Total Time considers both the creation and instantiation times. Complexity is measured according to the number of files that developers have to manage and the number of lines of code generated to create the system. It is clear from the displayed results that only one file (i.e, the `owl` generated by the **Designer** component) is required in all DES in OnToJade approach, whereas the number of files varies in JADE/Java approach depending on the DES size.

Regarding the number of code lines, there is an incremental growth for both approaches. However, in all scenarios, for OnToJade the number of code lines generated is nearly to 20% compared with the manual approach. Moreover, the required number of lines of codes in OnToJade has no significant change while the number of agents are increasing. No matter the MAS size, developers only manage one `owl` le for generating the system with OnToJade approach. In contrast, the number of les for the system increases exponentially with respect to the DES size in the manual approach, since the developer should have to create a le for every class that will be in the system. Thus, the amount of les brings more tasks for developers as the number of operations will increase in creating, modifying, compiling, and running those les. The manual approach can be challenging even for an advanced developer.

In terms of time, even if OnToJade takes almost 50% more time than the manual approach to instantiate MASs. This is because it is not only pure codes but also agent behaviors and relations that are expressed using the OJ Language are incorporated in OnToJade. This takes more time to parse. However, the total time for OnToJade is less than the manual approach in all scenarios. This is due to the time to create files in OnToJade is less than 20% than the manual one.

We also conducted a test for automatic generation of larger size DES with OnToJade. We created five agents which have different behaviors, roles, and resources; Then, we replicate them to have 1000 agents in a large DES. As shown in Table 4.2, the complexity remains the same. Only one file is managed by the developer and few code lines are added. The time remains less than one minute even for such large size DES.

TABLE 4.2: Large Size DES Creation in OnToJade Approach

Number of managed files	1
Number of code lines	∼25
Time to create file(s)	45 sec.
Time to instantiate	3.766 sec.
Total Time	**<1 min.**

To conclude, the complexity of the system has insignificant impact on the time to generate the DES using OnToJade approach.It enables to create complex Digital Ecosystems using only one source ontology file. This clearly discloses that our approach save time to developers in the process of development and generation MAS-based Digital Ecosystems. Thus, OnToJade is made to simplify the MAS-based Digital Ecosystem development process. The displayed results back our conclusion as OnToJade is a better solution to address both time and complexity.

4.4.2 Agent Interaction Test

This test focuses on the interaction of *Agents* to show the readiness of the prototype in supporting agents interaction in various ways and also to check usefulness of OJ Language while agents are exchanging messages. For the purpose of this test, three categories of agents are declared (i.e., *Requester, Provider, Idle*); three types of requests are defined (i.e., content, processing, support); and three DESs are created with different specifications as shown in Table 4.3

TABLE 4.3: Experimental protocol for Agent Interaction Test

DES	*Agents*	*Roles*
DES1	**A**	Requester
	B	Content Provider
DES2	**A**	Requester
	B	Content Provider
	C	Content Provider
	D	Processing Provider
DES3	**A**	Requestor
	B	Requestor
	C	Content Provider
	D	Content Provider
	E	processing Provider
	F	processing Provider
	G	idle

1) Unicast Interaction

Unicast refers to one-to-one communication between two agents in the system. The aim of this test is to assure that an agent can occur interaction any other agent in the system for message exchanging, sharing , and collaborating for resources. Based on the experimental protocol given in the above table (Table 4.3), Figure 4.16 displays the interface that a unicast communication between the two agents (*Agent A and B*) of DES1 has made successfully. The edges between the agents and the Moderator Agent show the subscription of the agents to the Moderator to join to the system. With this test, we also show that a unicast interaction is effected directly between two agents without need of passing through the Moderator Agent. In terms of OJ syntax, the unicast communication is expressed as follows:

***OJ instructions on Agent A (as Requester)*:**

```
AgentA action "SEND REQUEST @AgentB":string
RequestWaker action "SEND REQUEST @AgentB":string
```

***OJ instructions on Agent B (as Provider)*:**

```
AgentB RequestServer action "SEND INFORM @sender RESOURCE":string
action SEND INFORM @sender RESOURCE
RequestServer reactsTo "REQUEST":string
```

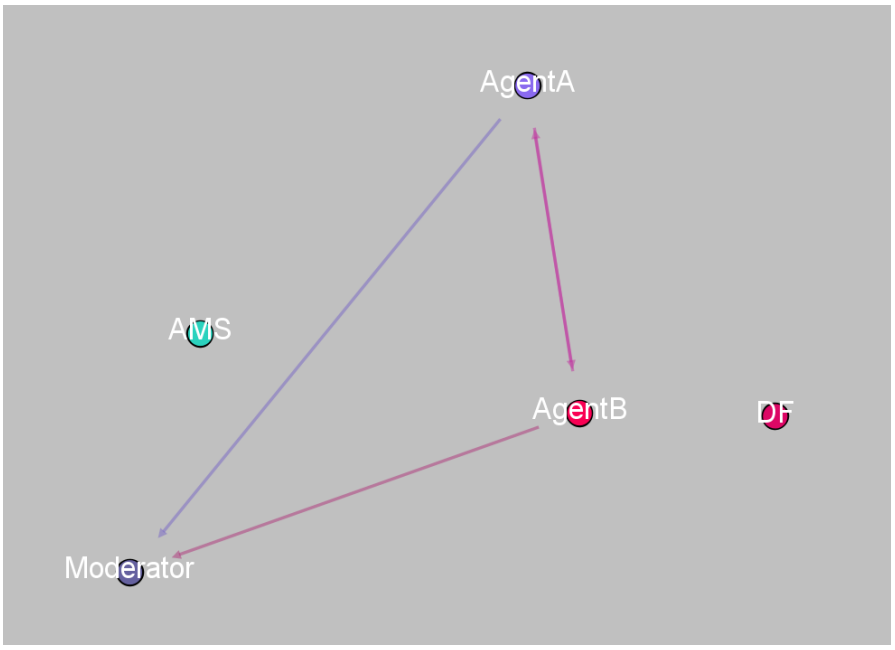

FIGURE 4.16: Unicast communication in DES1

The program also outputs the following information in the console to confirm our analysis above:

```
Agent AgentA launched !  Got 0 rule(s), 0 variable(s), 1 behaviour(s) and
0 resource(s) path(s) !
Agent AgentB launched !  Got 0 rule(s), 0 variable(s), 1 behaviour(s) and
1 resource(s) path(s) !

Moderator trace:  new agent asks to join the platform, name is:  AgentA
Moderator trace:  new agent asks to join the platform, name is:  AgentB

AgentB Trace:  Sending message(s) to:  Moderator, performative is SUBSCRIBE,
content is C:/Users/Pictures/Anglet.png
AgentA Trace:  Sending message(s) to:  AgentB, performative is REQUEST, content
is null
AgentB Trace:  message received from AgentA performative is REQUEST, content
is null
AgentB Trace:  Sending message(s) to:  AgentA, performative is INFORM, content
is C:/Users/Pictures/Anglet.png
AgentA Trace:  received a message from AgentB, performative is INFORM, content
is C:/Users/Pictures/Anglet.png
```

From the displayed graph and the console, two agents can directly communicate after their requests to join the system have got acceptance by the Moderator Agent as shown in the graph (Figure 4.16).

2) Multicast Interaction

The second experiment in this part checks a multicast communication among agents in the system. Multicast is a group communication where an agent is interacting to a group of agents (i.e, one-to-many communication). As shown in Figure 4.17, all agents must get registered by Moderator agent before occurring any communication. Those agents which have resources should be first registered by the DF JADE Agent of the platform as Agent B and C did. Agents that have not been registered to the DF JADE Agent cannot be contacted. These processes are expressed below using the OJ language besides the displayed graph (Figure 4.17).

Thus, in order to effect this type of communication, it is a must for an agent to get registered by the DF JADE Agent (below we show the OJ instructions in (1)). Thus, *Agent A* asks first the DF JADE Agent for agents that match the service type it is interested on (corresponding OJ instructions are marked as (2) below), DF JADE Agent responds that *Agents B and C* offer that service and then *Agent A* contacts them (OJ instructions in (3)).

OneShotbehavior (1)

```
RegisterToDF action "REGISTER AS "content-provider""
ProcessingRegistration "REGISTER AS "processing-provider""
```

Wakerbehavior (2)

Agent A action ”SEND REQUEST @TYPE:content-provider”

```
RequestWaker action "SEND REQUEST@TYPE:content-provider":string
```

ProcessingRequest action ”SEND REQUEST @AgentD RESOURCE WHERE NAME EQUALS ”hello.jar””

Cyclicbehavior (3)

```
ProcessingRequestServer action "SEND INFORM @sender RESULT OF EXEC MSG.CONTENT"
ProcessingRequestServer reacts to "REQUEST":string
```

```
RequestServer action "SEND INFORM @sender RESOURCE":string RequestServer reactsTo
"REQUEST":string
```

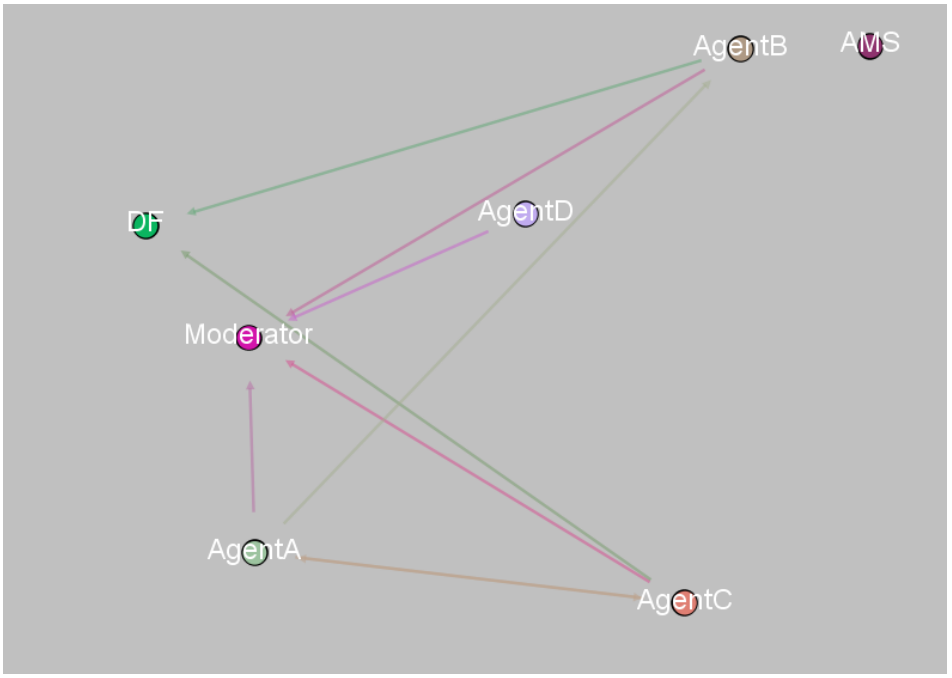

FIGURE 4.17: Multicast communication in DES2

The console output details for multicast communication is found in Appendix A.1. The graph as well as the console results clearly shown that all agents must be first registered by the Moderator Agent. The important observation in such type of communication is the involvement of the Moderator Agent to facilitate multicast interactions. Every communication passes via the Moderator Agent which was not the case in unicast communication.

3) Broadcast Communication

The third test is to assess the support of broadcasting a message to all agents in the system. This type of communication is to mention a single agent interacts to all agents. Figure 4.18 shows that an agent can communicate simultaneously to a number of agents. Not only this, there can be parallel communication among agents. For example, while *Agent A* interacts with the rest of the agents in the system, *Agent B* can also do that at the same time. In fact, all agents must be first recognized by the Moderator Agent and registered by the DF JADE Agent. A sample OJ Language syntax for broadcast communication is provided as follows:

```
RegistertoDF action "REGISTER" AS "content-provider""
Processingregistration action "REGISTER" AS "processing-provider"
Agent A RequestContents SEND REQUEST @ALL STRING ".jar"
Agent B RequestProcessing SEND REQUEST @ALL RESOURCE WHERE NAME EQUALS "hello.jar"
ProcessingRequestServer reactsTo "REQUEST"
ProcessingRequestServer action "SEND INFORM @sender RESULT OF EXEC MSG.CONTENT"
RequestServer reactsTo "REQUEST"
RequestServer action "SEND INFORM @sender RESOURCE"
```

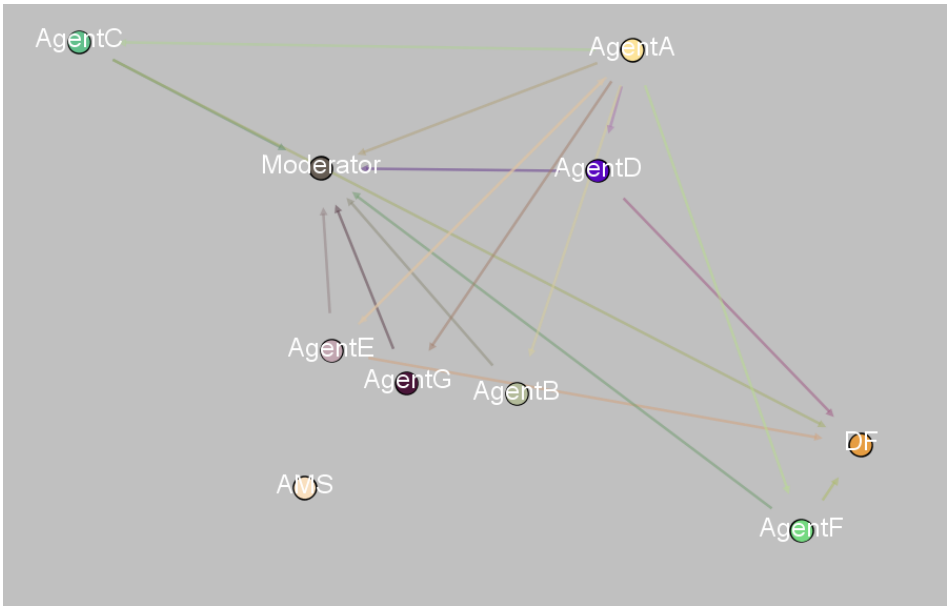

FIGURE 4.18: Broadcast communication in DES3

In Appendix A.2 is shown the detail console output. The results indicated that an agent can have communicate with two or more agents at a time and also possible to send messages to all agents.

To conclude, the test results and the displayed graphs show that OnToJade supports the three kinds of communication. It is also noted that the involvement of special agents is important to facilitate the smooth communication among agents and for proper coordination and management of the system.

4.4.3 Test for Comply System Level Rules

The last test is to validate the role of the Moderator Agent in terms of keeping the system level rules for maintaining consistency of the Digital Ecosystem.

The objective of this test is to validate that the system level rules are followed by each participant and also to check the role of the Moderator Agent in terms of keeping the system as consistent as possible. We introduce system level rules mainly: Rule for minimal number of agents, Rule for contribution, Rule for interaction, Rule for collaboration. These rules are integrated through the Moderator Agent. We created different DES at each level of the test.

Rule 1: minimum number of agents.

In a Digital Ecosystem, there must be at least two agents. For this test, we created a system with only one *Agent*. When the Moderator Agent observed that the amount of agents in the system was not enough to function as a DES, it automatically ceases the DES after a short delay. Here is below the trace of the Moderator Agent in this situation:

```
Moderator trace:  System currently running with a number of agents running
on the platform (1) inferior to the constant "noOfAgents", system will shutdown
automatically if no agents comes to the platform!
```

This result indicates that there must be at least two agents so that the DES exists and the platform supports it.

Rule 2: minimum number of contents. For this test, we created an ontology with two different agents, *AgentA* and *AgentB*. *Agent A* is linked to a resource folder; whereas *AgentB* has no any resource. The Moderator realized that *AgentB* does not fulfill the minimum threshold for resources to stay in the system and it automatically add it to the list of agents to kill. The console trace below with Figure 4.19 for the Rule2 test gives more detail about the test result of Rule 2.

```
Moderator trace:  new agent asks to join the platform, name is:  AgentA
Moderator trace:  new agent asks to join the platform, name is:  AgentB
AgentA Trace:  Sending message(s) to:  Moderator, performative is SUBSCRIBE,
content is C:/Users/Pictures/Images/screenshot0.png
Moderator trace:  AgentB have a number of resources inferior to the constant
"contentMinimumAmount" name added tokickList!
```

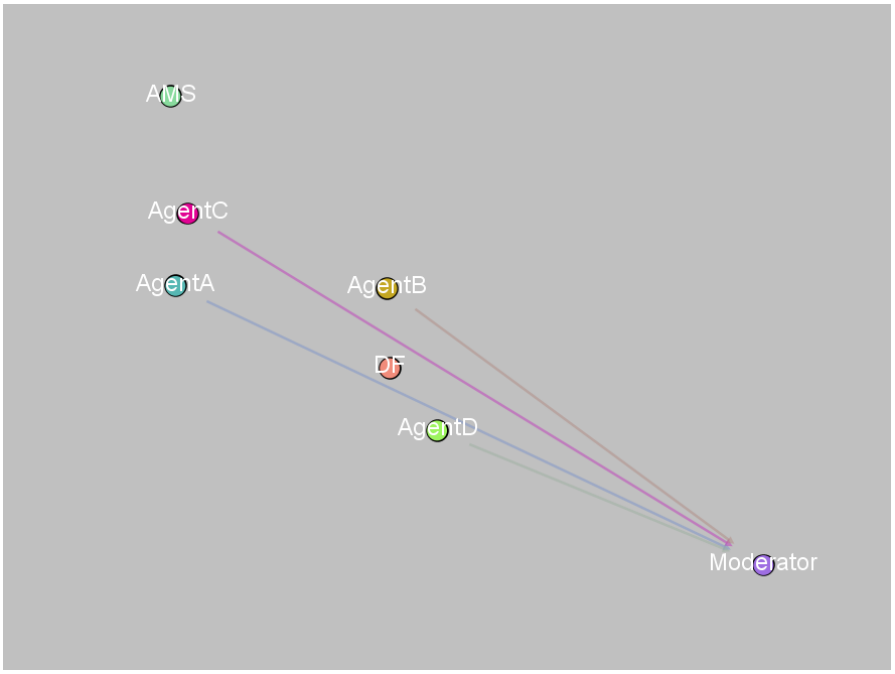

FIGURE 4.19: Test Result for Rule 2

The result tells that each agent in the MAS should contribute so that mutual benefit could ensure. otherwise the Moderator Agent will be forced to take actions. This is important to keep equilibrium of the systems and punish free-rider agents.

Rule 3: TimeToQuit. This test is based on the *timeToQuit* variable and shows that the Moderator Agent will automatically kill an agent that does not interact with its environment, contribute, and collaborate in a given time frame. For the purpose of this test, we created a DES containing four different agents. *AgentA* sends a message every three seconds to every agent on the platform; *AgentB* has resources but does not share them; *AgentC* declares its resources and *AgentD* sends a message every three seconds to every agent but has no resource. According to the rules for interaction, contribution, and collaboration, the Moderator Agent did not kick*AgentA* and *AgentB*. However, it puts *AgentB* and *AgentD* to KickList. Though *Agent B* declared its resource, it did not interact to other agents until its *timeToQuit* reach zero. *AgentD* did not meet the minimum amount of resource to stay in the system. In addition to Figure 4.20, the console trace in Appendix A.3 provides the details about this test.

To summarize, the tests on rules has shown it is possible to define and implement different kinds of rules at system and individual agent levels to ensure the benefits of all participants and for the balance of the system. With this, the results displayed that

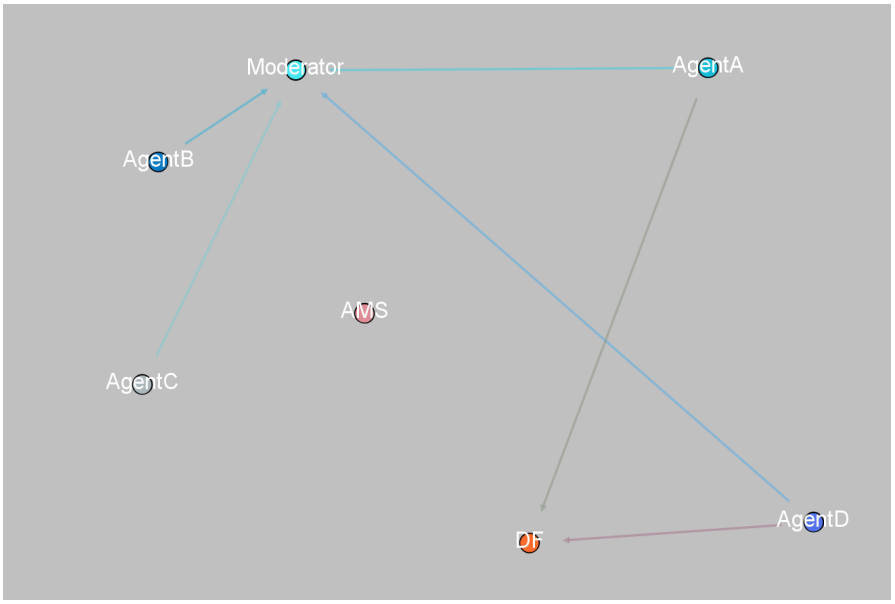

FIGURE 4.20: Test Result for TimeToQuit

the Moderator Agent is capable of enforcing system level rules. And at last, any agent which is not ready to comply system level rules cannot survive in the system.

4.5 Conclusion

We described Onto2MAS, an ontology-based framework, aimed to support automatic generation of MAS-based systems. Onto2MAS integrates designing, generation, and deploying phases of such complex systems in an easy to use tool (called OnToJade) implemented with well known open-source utilities (e.g., Protégé, JADE, Jena). Onto2MAS also provides a language, called OJ, to help the designer in the process of the specification of end-user requirements and rules that will govern the MAS-based Digital Ecosystem. We conducted experimental tests, with OnToJade that demonstrate its efficiency and effectiveness on automatic generation of MAS, without demanding special programming skills. The experimental result is very encouraging that the proposed framework is practical in handling complexity, minimal file size, and less time for generating a system. Moreover, the framework is designed to support to any complex systems which could be developed from MAS basis. In the next chapter, we will show how our framework supports the development of Digital Ecosystem for multimedia contents sharing and collaboration.

Chapter 5

MMDES: MultiMedia Digital Ecosystem

Currently multimedia contents dominate the information exchanged in Internet, particularly through social networks. Each actor in Internet (individuals, enterprises, territorial communities, etc.) becomes producer and consumer of contents. Development and increasing demand of multimedia data offer producers/consumers more choices and opportunities for sharing in collaborative environments. Nevertheless, social network platforms and other traditional collaborative environments present limitations regarding content selection and collection, categorization and classification, aggregation, linking and interoperability, usage control and privacy. In this context, we propose a MultiMedia-oriented Digital EcoSystem (MMDES) as a new environment of collaboration, sharing, and computing of multimedia content and applications ensuring the benefits of all its participants.

In this chapter, we first evaluate social media and collaboration platforms. Then, we present the specific requirements and objectives of MMDES. Finally, we demonstrate in detail how the ontology MAS2DES-Onto and the framework Onto2MAS, previously proposed, are used to generate MMDES, a Digital Ecosystem for the specific domain of multimedia sharing and collaboration.

5.1 Social Media and Collaborative Platforms

With the rapid evolution of social media, supported by the vertiginous evolution of Internet and Web 2.0, many communication, information sharing, and collaboration platforms have emerged for different relationships among people [118]. These media offer users the opportunity to create and share massive heterogeneous data. Social media encourage, support, and enable people to share their information and knowledge easily and effectively through different mechanisms [119]. They encourage interaction, participation, and socialization amongst users of a community [120].

A wide variety of characteristics and capabilities have been defined for social media in the literature:

- **User-generated content**. Creation of the content is one of the characteristics of social media [121]; users are no longer just simple readers, but rather they can contribute, collaborate, and participate in content generation [122].
- **User-to-user communication**. Connectivity is the feature of social media that enables people easily to stay connected with each other in a real-time and in a global base [123].
- **Networking**. Building a community of users is another characteristic of social media [122]. It has enabled people with common interest gather together in an online space, locate each other, share their profiles, develop relationships, and transfer their knowledge and experiences.
- **Multimedia-oriented content**. Social media provide opportunity for users to publish and share their own created multi-media content in different forms such as text, image, audio, video, and other formats in an interactive and easy way [124, 125]. As a result, millions of multimedia contents have been exchanged among individual users since social media platforms get launched. YouTube, Flicker, and various Podcast services are examples of social media for multimedia sharing which allows people to share variety of video and photo files with different subjects [126]. These media do not require high technical proficiency or special programming skills to use [127].

Multimedia is determined as important media to share users experience and in transferring knowledge in social media [128]. It is obvious that multimedia objects can motivate individuals to publish and share different events [129]. It is estimated that 80% of the shared information on the Internet is with multimedia formats [130]. The vast amounts of multimedia contents are coming from different sources with diversified representations. Though these media provide collaborative environments, there are constrains in accessing or using social media application tools [131]. We present these challenges in the following section.

5.1.1 Challenges of Social Media Platforms

There are emerging challenges about social media platforms as identified in [20]. We discuss some of these challenges as follows.

1. Selection and collection of contents

Even social media platforms enabled users to share and publish large quantities of multimedia contents, these platforms only have an interface mainly for sharing resources with simple search engine [49]. The most common way to access contents from social media is to use the API (Application Programming Interface) offered by the service-provider. These APIs are often restricted to execute within the confines of specific social network platforms [132, 133]. Social search engines are also emerged to access data from those media. However, these search engines crawl social networks and index the available content based on text (i.e., they are keyword-based social search engines) [134]. Thus, the amount and diversity of multimedia data shared through these enormous social media collections pose more parameters to consider towards efficient selection and collection. A major challenge, therefore, is in finding solutions to open up social media platforms to allow cross-platform selection, exchange, and usage of multimedia resources.

2. Classification and categorization of contents

The vast amount of social media generates massive amounts of multimedia contents on a daily basis, often diversified and heterogeneous. It is hard to discount any of these sources of information, and yet, there is a vast amount of redundancy and noise that has to be overcome in order to extract relevant and actionable information from them. Additionally, these sources generate content that differ in size, frequency, expertise, and

relevance. They have their own syntaxes and terms for representing data. This makes increasingly difficult to glean true and useful information from them. Moreover, they are not also classified according to place, time, and subject and no easy discovery of semantically related resources [135]. Automatically categorizing and compressing important contextual information from these sources is crucial for tasks such as document classification, categorization, and summarization [136].

3. Aggregation and interoperability of data and services

In social media environments, users are wondering how they could assemble the information across several networks. Data aggregation is any process in which information is expressed in a summary form for purposes such as reporting or analysis. This process allows to put the needed contents into one space. It helps to mine and organize correlated information from many different sources. Shared multimedia contents existing in social media are created by different users with different background, the data being formatted following different standards, and using different kinds of techniques. Given the different varieties of sources, the task of integrating and understanding information from these heterogeneous sources is both a difficult and important challenge. To represent heterogeneous data across social media, a unified schema is required. There is no common description technique in terms of representations of structure and content those are the basis for deriving links across media [137]. In addition, the current social media has been designed for human consumption, but not for automatic processing by machine [137].

4. Extraction of new knowledge

Knowledge discovery is the process of automatic extraction of interesting and useful knowledge from large data set. Social networks contain immense knowledge through their users. The knowledge extracted about users, resources, and their links is important to improve data accessibility, management, and exchange [138, 139]. This would be more important when the volume of data and knowledge continues to accelerate [49]. By correlating knowledge of different users, individuals can gain new understandings, discover new relations, and produce more knowledge. However, existing social media platforms could not support services for efficiently utilizing individual knowledge to the superior benefit of their communities [137].

5. Support for Collaboration

By working together, users can accomplish what none of them could alone. Collaboration extends benefits. It helps participants to work and support one another, to share applications, and to interact almost as though they were in the same place. Users are producing and sharing constantly growing amount of multimedia resources on the web [140]. But they do not possess all the right skills and tools to perform annotation, feature extraction, content description, coding and decoding. Thus, they need support from others. However, there is no well-developed means to execute such tasks in a collaborative way among social media users. To the best of our knowledge, there is no comprehensive platform that delivers multiple methods of collaborating, sharing, and managing services and expertise in the social media world.

6. Data and usage control

The last challenge is from data and usage control point of view. Users are not in control of their data and usage, which leads to loss of trust among them, drop of privacy, and hinder them from freely sharing and collaborating resources.

The recent advance in computing and networking technologies has fueled the emergence of social media with requirement of better collaboration environment [120]. In order for these platforms to be truly useful and effective, they must be able to operate in a collaborative fashion. There are distinct distributed and collaborative environments which allow two or more participants to communicate, coordinate, and assist each other. In the following section, we present different collaborative environments that support social media and others.

5.1.2 Distributed and Collaborative Environments

Traditional collaborative environments support communication among collaborators, however, they present limitations regarding several aspects [20, 32].

In **client/server environments**, a client requests a content or an execution of services from a centralized server [141, 142]. Cooperative processing of multimedia data and applications would be really restrictive due to the full dependency on the central coordinator which might not satisfy the needs of all participants at the same time.

In **peer-to-peer (P2P) environments**, participants could play both roles (requester and provider) without centralized control and authority [143]. However, participants may face difficulty of managing their interactions as large number of peers are frequently joining and leaving P2P environments [144]. In addition, as stated in [145], many participants in P2P are freeloaders who take advantage of the available resources but do not contribute in return. This would have a negative effect on sharing resources and also discourage cooperation among participants.

Grid environments provide distributed computing power infrastructure in very large scale heterogeneous systems [130]. Each participant is autonomous in grid environment to determine about its resource sharing and collaboration [32, 146, 147]. Grid relies heavily on dispersed data management and connectivity environments to handle challenging scientific and engineering computations. This makes users fear more about loss of control of data and systems [148].

In **cloud environments**, underlying infrastructure and applications are abstracted and offered as a set of services. One disadvantage of clouds is the implicit dependency on the providers as their resource usage and activities rely on them. In addition, privacy becomes an issue since providers might outsource some sensitive and confidential data and applications to third parties [149–151].

Though the above environments are supporting sharing and publishing data, they have several constraints to cope with multimedia-based needs in terms of collaboration, publishing, and sharing [20, 22]. Moreover, most shared multimedia applications use the client/server model in which data and services are stored on and routed through dedicated servers [152].

To address these shortcomings and meet the increasing interest in sharing appropriately multimedia resources, we provide a collaboration environment based on the concepts of Digital Ecosystems. As the underlined platform, Digital Ecosystems overcome the limitations of traditional collaborative environments. We call our collaborative environment, MMDES (**M**ulti**M**edia-oriented **D**igital **E**co**S**ystem) [20, 22]. In the next sections, we detail our proposal to its implementation stage.

5.2 MMDES: New Platform for Collaboration and Multimedia Sharing

5.2.1 Overview of MMDES

MMDES is a special type of Digital Ecosystem which aims at creating a digital environment for interested group of entities that support cooperation and promote knowledge sharing to main mutual benefits. MMDES can be taken as a new way to handle collaboration in a distributed and heterogeneous environment. It focuses on sharing multimedia services and resources such as multimedia contents, multimedia processing and application, cache, and storage capabilities among participants in a win-win kind of interaction. MMDES provides several features and advantages from user and system perspectives.

From system perspective, MMDES addresses challenges related to select, enrich, extract semantics, and combine multimedia resources from different sources so to come up with a collective intelligence for effective and better recommendation, negotiation, notification, etc. From user perspective, it allows users to define their profiles and preferences, to publish or keep locally their resources, to have the benefits from collective knowledge for better collaboration, and to have control over usage of their resources according to predefined rules with regards to user preferences and interests. This aspect of MMDES addresses the challenges linked with collaboration and ownership of resources mentioned in the previous section (Section 5.1.1).

MMDES general environment is graphically presented in Figure 5.1. It has three levels: **Participant**, **Community**, and **Environment**. We call *Participant* to a member in MMDES. A participant in MMDES can be composed by agents that can play specific roles, compose a set of resources to offer and share, define a set of preferences and rules to how and who can use its resources, and a representation of its experiences conforming its knowledge. MMDES Participants can be grouped into *Communities* according to common preferences and with resources associated to a common domain. Communities can be considered as small ecosystems since they can define common preferences and rules, and generate a collective knowledge in the context of a specific domain. Individual participants can assume roles in a community or in MMDES. Individual and communities are subsets that made up MMDES *Environment*. Thus, MMDES objectives and requirements will be defined next to support this environment at various levels.

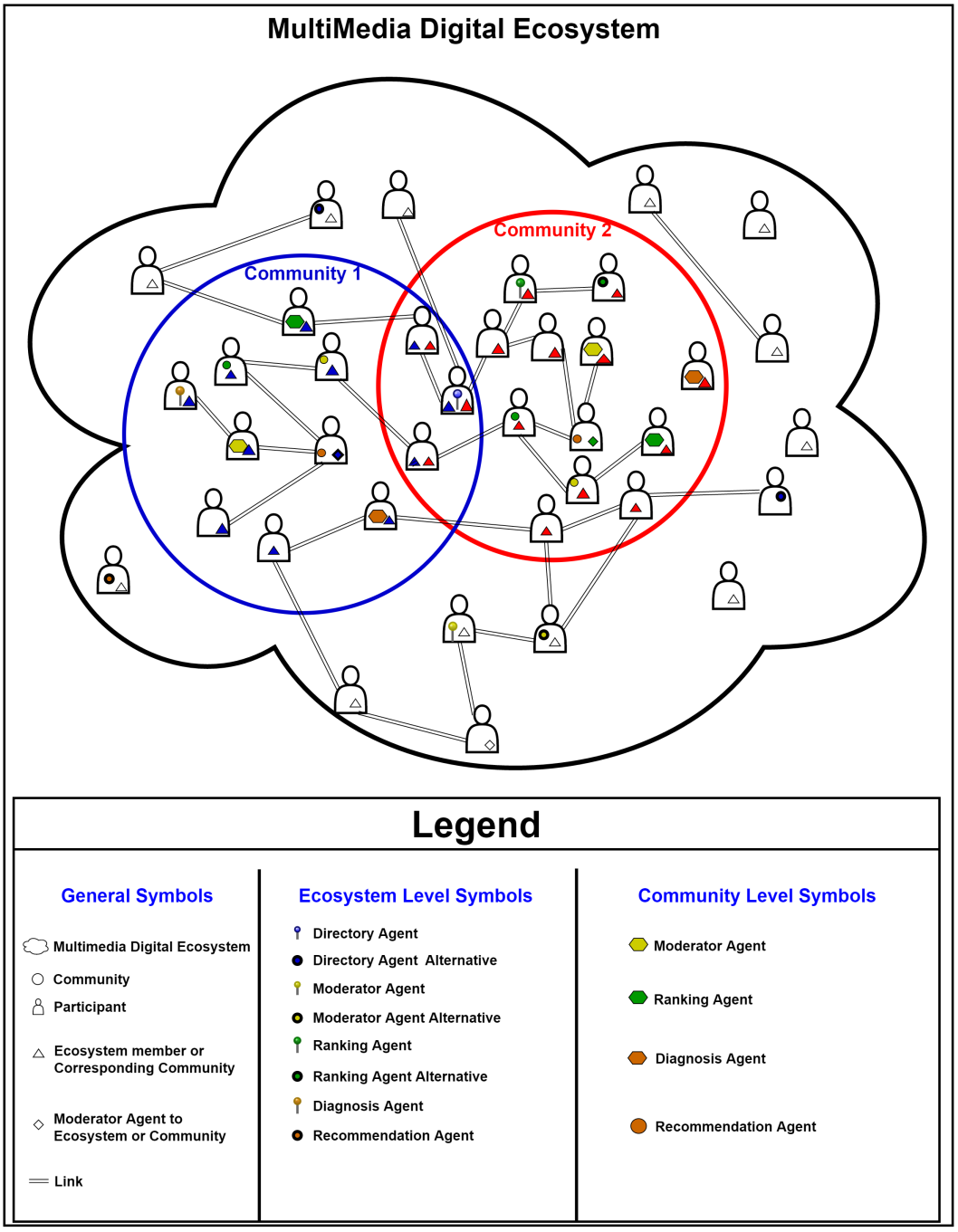

FIGURE 5.1: MMDES Environment

5.2.2 Objectives of MMDES

Various platforms and infrastructures are available to support communication, coordination, and cooperation among users. As mentioned previously, Digital Ecosystems are attracting more and more interests from a number of communities and have emerged as a new collaborative environment thanks to interesting characteristics in supporting collaboration and cooperation [32, 62, 153, 154]. It is an appropriate means for multimedia sharing and collaboration that transcends traditional collaborative environments [**?**]. As it is stated in Chapter 2, Digital Ecosystems have characteristics which attract our attention for various reasons. MMDES inherits the characteristics of Digital Ecosystem to realize the following four main objectives:

- **Promoting Interaction and Collaboration**. Interaction and collaboration among participants and their environment are fundamental for the survival and functionality of a Digital Ecosystem and its participants. In this regard, MMDES encourages all participants to interact and collaborate. Interaction and cooperation depends on preference and interest of participants. The collaboration is lined based on their mutual benefit. MMDES represents an interactive community: (i) participants interact to reach common goals, to find common subjects of interest, and to share resources; and (ii) interaction is the base for sharing and collaboration as well as discourage free-riders and selfish participants.

- **Autonomy of Individuals in Resource Management**. MMDES guarantees to participants the full control and usage (i.e., *how* and *who* providing access to each participant resources) on their resources while meeting their requirements and interests. It ensures resource ownership and control, allowing autonomy, and self-organization of participants. Thus, participants can determine their own requirements and interests; and each participant can determine how and who can access its resources. In MMDES, there is no fixed role assignment. As a result of this, each participating entity has dual roles of providing and consuming resources and services. Based on its role, every participant has a chance to own and manage its resources.

- **Equilibrium**. MMDES guarantees mutual benefit among all participants. This would be achieved by measuring the contribution of each participant. Keeping balance in terms of resource provision and consumption is a responsibility of everyone in the ecosystem. Their contribution is crucial for the survival and reliability of the ecosystem. To make certain this, MMDES establishes weighting of participation. It is necessary to automatically measure the level of participation of each individual according its role in the ecosystem and based on some metrics.

- **Promoting Shared Knowledge**. MMDES ables to create a shared knowledge from participants based on their interaction, behavior, resource, and preference. The ecosystem will support them to learn and profit from collective knowledge of the community. Thus, MMDES updates constantly participants with mechanisms of recommendation. This allows to analyze participant preferences, performance in the ecosystem (level of participation), and behavior (e.g., favorite contents, members to whom frequently exchanges, interacts, shares) to generate recommendations to improve participant experiences, integrate to related communities, create a new community, or even suggest a change of behavior, such as recommend to request a resource to a participant different from the usual one, or suggest to free a resource that is sub-utilized, in order to keep the ecosystem balance.

To sum up, this new environment exceeds the existing collaborative environments by providing a win-win interactive collaborative environment for better management of multimedia contents while dealing with individual preferences and constraints. The next step is to develop MMDES according to previously proposed Onto2MAS Framework, which is composed of three main components: **Designer**, **Generator**, and **Deployer**. The **Designer** component supports for defining the ontology of MMDES from MAS2DES-Onto, that describes the concepts of MASs and Digital Ecosystems. The **Generator** component creates MMDES agents that provide system wide support and others for management of multimedia resources sharing and collaboration. The **Deployer** is in charge of creating an application and attain configuration. Hence, in the next section we detail the development process of MMDES.

5.3 Design Process of MMDES

The *Design process* of MMDES is achieved through MAS2DES-Onto (an ontology that describes all the concepts and rules to follow when instantiating the Digital Ecosystem) to derive an ontology that copes with specific needs and requirements of MMDES. Next subsections describe the main concepts and generic rules of MMDES.

5.3.1 MMDES Derived Ontology

Creation of an ontology is the first step of the Design process according to Onto2MAS framework to represent MMDES. We derive the MMDES ontology shown in Figure 5.2 from MAS2DES-Onto. From the five modules of MAS2DES-Onto, we took the following main concepts: *Digital Ecosystem*, *Agent*, *Profile*, *Resource*, *Rule*, *Role*, *Knowledge*, *Action*, and *Interface*. In addition to the concepts from MAS2DES-Onto, we also provide instances of some of these concepts which are new and identified for the proper implementation of MMDES. As the final goal of this work is to provide a Digital Ecosystem for multimedia sharing and collaboration, we formally define and present each of the concepts in this context as follows.

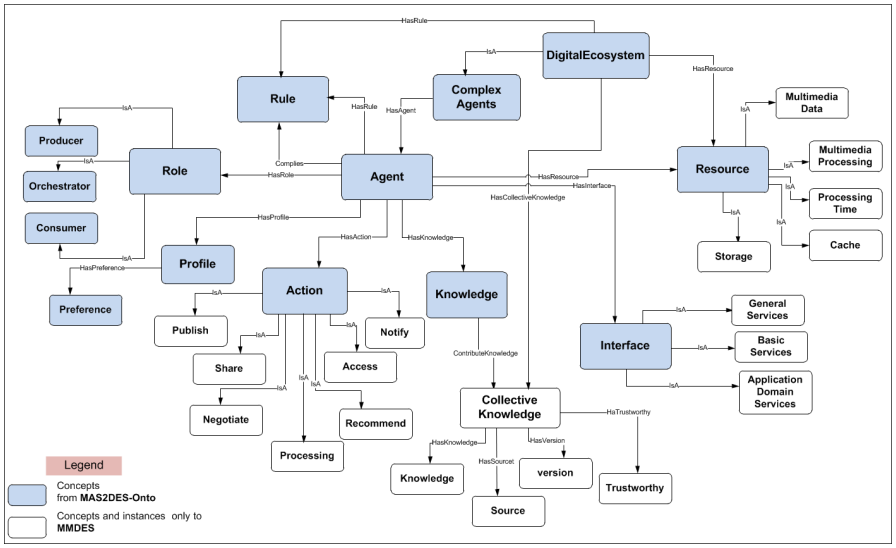

FIGURE 5.2: MMDES Concepts

Digital Ecosystem concept. A Digital Ecosystem consists of four main elements: Participants, Resources, Rules, and Collective Knowledge. Def. 5.1 formally defines the core concepts.

Definition 5.1. Digital Ecosystem. *A Digital Ecosystem, denoted as MMDES, is represented as a 4-tuple, DES=(PARTICIPANTS, RESOURCES, RULES, COLLECTIVE_KNOWLEDGE),*

where:

- *PARTICIPANTS is a non-empty set of participants that can be AGENTS or COMPLEX AGENTS, in DES; such that* $PARTICIPANTS = \{p_1, p_2, p_3, \ldots, p_n\}$ *and* $|PARTICIPANTS| \geq 2$; p_i *is an AGENT concept or a COMPLEX AGENT concept;*
- *RESOURCES is a set of resources that are available in DES to* $\forall p_i \in PARTICIPANTS$, *such that RESOURCES* $= \{r_1, r_2, r_3, \ldots, r_n\}$ *and* r_i *is a RESOURCE concept;*
- *RULES is a set of rules; such that* $RULES = \{r_1, \ldots, r_m\}$ *and* r_i *is a RULE concept and* $\exists pi \in PARTICIPANTS$ *that define* r_i *and should be respected by all participants of DES;*
- *COLLECTIVE_KNOWLEDGE is a representation of the set of knowledge extracted and aggregated from participants to create new values, such that*

  $COLLECTIVE\_KNOWLEDGE =$

  $\bigcup_{\forall pi \in PARTICIPANTS} (KNOWLEDGE_{pi}) \oplus \prod_{\forall pi \in PARTICIPANTS} (KNOWLEDGE_{pi})$,

  where:

  $KNOWLEDGE_{pi}$ *is a KNOWLEDGE concept provided by* $p_i \in PARTICIPANTS$, $\bigcup$ *represents extraction, and* $\prod$ *represents aggregation.*

For example, MultiMedia Digital Ecosystem (MMDES) is a typical example for Digital Ecosystem as it has different participants from human users to different tools and application for processing, publishing, and sharing multimedia resources as the case in the documentary movie production scenario. There are human participants like Alice and other journalists, organizations like WHO, and multimedia tools and applications as a component of the system. These users have different interest and owns resources about Ebola epidemic. Resources are shared among themselves depending on defined rules for usage and manipulation. In addition, these participants collaborate in content provision, expertise and processing services to produce the planned documentary and finally publish and share the documentary. From the aggregated multimedia contents, journalists and other stakeholders could learn from their usage experiences which is represented and extracted as a collective knowledge.

A) AGENT concept

AGENT represents a participant which is the basic play maker of MMDES ecosystem. Each agent is associated with a set of preferences and rules in multimedia contents that shape its behavior within MMDES. An agent can be associated with several roles (e.g., consumer, producer, orchestrator) in MMDES. Each role has a set of associated privileges. The default role for a new individual is *Beginner*, whose unique privilege is to apply to become a member of MMDES. There exist special roles in MMDES to designate *Authorities*, who act over the whole ecosystem (or a particular community) with special privileges. We formally define the concept agent in Def. 5.2.

Definition 5.2. AGENT. *An agent is a representation of a participant in MMDES, denoted as "a" and* $a \in PARTICIPANTS$*, is defined as a 8-tuple representation,* $a=(a_{id}$*, PROFILE, AGENT_RESOURCES, AGENT_KNOWLEDGE, AGENT_RULES, ACTIONS, ROLES, INTERFACES),*

where:

- a_{id} *is a unique identifier of "a";*
- *PROFILE is a set of information to describe the characteristics of "a" such as {location, interest, preferences};*
- *AGENT_RESOURCES is a set of resources that "a" owns and/or manages; such that* $\forall ars_{aid} \in AGENT\_RESOURCES$*,* $ars_{aid} \in RESOURCES$*, and* ars_{aid} *is a RESOURCE concept;*
- *AGENT_KNOWLEDGE is a set of knowledge about a's fact and experience; such that* $\forall k_{aid} \in AGENT\_KNOWLEDGE$*,* $k_{aid} \in COLLECTIVE\_KNOWLEDGE$*, and* k_{aid} *is a KNOWLEDGE concept ;*
- *AGENT_RULES is a set of rules that states a collaboration and sharing policies; such that* $\forall aru_{aid} \in AGENT\_RULES$*,* $aru_{aid} \in RULES$*, and* aru_{aid} *is a RULE concept;*
- *ACTIONS is a set of possible tasks that can be carried out by "a"; such that* $\forall aac_{aid} \in ACTIONS$*,* $aac_{aid} \in$ *{publish, recommend, negotiate, notify, promotion, access, processing}; and* aac_{aid} *is an ACTION concept;*
- *ROLES is a set of capacities that "a" can play; such that* $\forall aro_{aid} \in ROLES$*,* $aro_{aid} \in$ *{producer, consumer, orchestrator} and* aro_{aid} *is a ROLE concept;*
- *INTERFACES is a description of several services offered by "a", denoted as interface and interface* $\in$ *{general_services, basic_services, application_domain_services}, is represented as 6-tuple,*

interface={NAME, TYPE, MESSAGE, OPERATION, BINDING, PORT}, where:

- *NAME is the set of names of services;*
- *TYPE describes the set of data type used by services;*
- *MESSAGE is a definition of set of data being communicated for each service;*
- *OPERATION describes set of actions supported by services;*
- *BINDING describes how a set of operations is involved; and*
- *PORT defines the set of addresses or connection points to services.*

ACTION is a concept which represents the set of actions that an agent can provide or execute on multimedia resources. Note that ACTIONS refer to a task that an agent is executing to survive and achieve its goal in the ecosystem. Some possible actions are: (i) Publish, is an action of presenting multimedia resources, applications, and services for access; (ii) Recommend, is an action for suggesting others to discover resources and services that interest them; (iii) Negotiate, is an action of establishing an agreement for sharing and collaboration; (iv) Notify, is an action for informing about multimedia resources, applications, and services according to their interest; (v) Promotion, is an action of advertisement for attracting other participants; (vi) Access, is an action of using and retrieving resources and services of others according to their usage rules; and (vii) Processing, is a processing to manipulate multimedia contents which can be file compression, merging, multimedia conversion, face recognition, etc.

ROLE refers to the role of an agent in the interaction with other agents. Examples of roles are: (i) Producer, for an agent that delivers multimedia resources and services; (ii) Consumer, for an agent that requests and accesses multimedia resources and services; (iii) Orchestrator, is a role of an agent that play coordination tasks.

KNOWLEDGE represents of set of facts, experience, and history of an agent about itself, other agents, and the environment.

INTERFACE represents the way to access, manipulate, and display resource services/applications, such as general services, basic services, and application domain services. An interface characterizes a means for interaction, collaboration, and service provision. We distinguish three kind of services: (i) General services, representing common facilities of the ecosystem which are oriented towards end-user applications; (ii) Basic

services, which represent domain-independent interfaces; and (iii) Application domain services, which are oriented towards specific application domains.

Definition 5.3. COMPLEX AGENT. *A Complex agent is a participant in DES which is composed by two or more agents, denoted as ca and* $ca \in PARTICIPANTS$*, is defined as a 2-tuple representation,* $ca=(ca_{id}, AGENTS)$*,*

where:

- ca_{id} *is a unique identifier of ca;*
- *AGENTS is a set of agents composing ca and* $AGENTS=\{a_1, a_2, a_3, \ldots, a_n\}$ *and ai is an AGENT concept.*

Example: many simple agents can cooperate to compose a more complex application. In the scenario provided previously about documentary movie production, there are agents performing automatic filtering of text files about Ebola disease, such as mail, reports, or news, or looking for movements in a video stream about the epidemic to assist the movie maker. Thus, different agents work together to solve problems related to filtering and categorizing of collecting data about Ebola in different countries in the infected region.

B) RESOURCE Concept

RESOURCE is a concept which represents the set of assets that an agent possesses and manages. It can represent a multimedia data, a multimedia processing/service, a processing time/period, a cache, and a storage facilities. Def. 5.4 formally defines this concept.

Definition 5.4. RESOURCE. *A resource, denoted as rs, and* $rs \in \{multimedia\_data, multimedia\_processing, processing\_time, cache, storage\}$*, is represented as a 5-tuple,* $rs=\{rid, , metadata, type, link, owner\}$*, where:*

- *rid is an identifier of rs, through which it can be accessed in MMDES;*
- *metadata is the set of features of rs, such as title, creator, subject, description, publisher, date, format, language;*
- *type gives an information about rs type;*
- *link indicates a point in MMDES by giving it a URL of rs; and*
- *owner states an owner of rs in MMDES.*

In MMDES, every participant comes to the ecosystem with some resources for sharing to others. Resources include: (i) **Multimedia data**, which refers contents that uses a combination of text, image, audio, video, or graphical objects; (ii) **Multimedia processing**, which is the capability to manipulate multimedia contents; (iii) **Processing time**, which is the amount of CPU time that agents can use for processing multimedia contents; (iv) **Cache**, to store copies of frequently requested multimedia data and process for quick access to improve overall performance of the ecosystem; and (v) **Storage**, is a sharable device used for storing multimedia data permanently for future use.

C) RULE concept

RULE refers to the set of conditions allowing to cope with the goals of the Digital Ecosystem and with the resource owners requirements. Thus, three types of rules can be defined in MMDES: (i) **SYSTEM RULES** are conditions governing the ecosystem to allow its members participate in the definition and configuration of MMDES; these initial rules have to be defined at the genesis of the ecosystem but can be modified in consensus among participants; (ii) **COMMUNITY RULES** are rules related to a group of agents which may be formed based on their common interest or goal; or the same rule that is defined by two or more agents can be considered as community rule (i.e., intersection or rules among agents) and (ii) **AGENT RULES** are conditions related to the owners of resources stating how to access and consume them. We formally define Rule in Def. 5.5.

Definition 5.5. RULE. *A rule, denoted as r and $r \in RULES$, is defined as a 3-tuple, r={rid, name, description}, where:*

- *rid is a rule identifier of r;*
- *name is a name of r; and*
- *description is a definition of r.*

For instance, there must be at least two participants to assume MMDES as Digital Ecosystems and every participant should interact to another participant in a defined time period could be taken as example of system level rules. As an example for community rule, participants may set a minimum storage and CPU capacity for those requesting a video contents. Copyright and fair use of multimedia contents can be set as a rule

by individual participants depending on their concern. In the context of documentary movie production, Alice may set usage rules about the documentary that it is only used policy makers of WHO. The documentary is not to be televised to the world as it contains strange images and videos about people of the infected areas.

As previously mentioned in Section **??**, we defined and experimented system level rules related to Interaction, Resource Contribution, and Cooperation. These rules are integrated in Moderator Agent of OnToJade. In addition to those three rules, we added two rules in MMDES as follows.

Rule for Comply Usage Rules of Participants

Owners of resources should have the authority to determine to whom to share and about the usage of shared resources. This could enable to protect any potential misuse. It also enforces participants to react responsively. Moreover, it brings trust among participants as trust is an important factor for the relationship between two entities. The trust landscape plays an important role in collaboration in order to protect rational users against malicious ones. Thus, participants must comply usage rules of resource owners to keep right, interest, and benefit of others. Then each participant will build respectful relationship and care about the contents shared to others. This rule is displayed in Algorithm 5.

Algorithm 5 Comply Usage Rules

while true **do**

  $Agent_i$ `Request_`$action_x$ $Agent_j$

  `if (`$Agent_i$ `COMPLY` $Rules_Agent_j$`) AND (`$Agent_j$ `COMPLY` $Rules_Agent_i$`) then`

    `(`$Agent_j$ `Provide_`$action_x$ $Agent_i$`)`

  `else`

    `(`$Agent_j$ `Reject_`$action_x$ $Agent_i$ `)`

  `end if`

`end while`

Rule for Contributing Knowledge for the Ecosystem

Collective knowledge is constructed from contributed knowledge of participants. This combined knowledge is required for building new knowledge for better decision and

collaboration. This collective ability is vital for the success of the ecosystem as it allows them to act intelligently. Participants contribute their knowledge to get the best return from the mass. Then, collected knowledge will be analyzed for community or domain-specific purpose so that effective collaboration could be realized. Algorithm 6 summarizes this rule.

Algorithm 6 Contributing Knowledge to the Ecosystem

while true **do**

$\forall Agent_i \in$ PARTICIPANTS

if $(AGENT\_KNOWLEDGE_{a_i})$ *related_to* $COLLECTIVE\_KNOWLEDGE)$ **then**

$(Agent_i$ CONTRIBUTE $AGENT\_KNOWLEDGE_{a_i})$

else

(*Add it to the alert list*)

end if

end while

D) COLLECTIVE KNOWLEDGE Concept

Collective Knowledge (CK) is the aggregated knowledge assets contributed by participants of MMDES about shared resources, usage, and experiences [21]. It is used for improving service provision of participants so that they could benefit more from MMDES. Def. 5.6 states this concept.

Definition 5.6. COLLECTIVE KNOWLEDGE. *A collective knowledge, denoted as ck and ck* $\in$ *COLLECTIVE_KNOWLEDGE, is represented as a 5-tuple,*

ck=$\{KNOWLEDGES, S, V, t, K' \}$, *where:*

- *KNOWLEDGES* ($\sum k$) *is the summation of pieces of knowledge k in the form of RDF Triple (s,p,o) subject, predicate and object;*
- K' *the previous version of the knowledge K (K is evolved from the previous knowledge* K'*);*
- *S is the set of sources of the pieces of knowledge;*
- *V is Version of the CK; and*
- *t is trustiness of the source /trustworthy.*

5.3.2 MMDES Ontology File

As mentioned in Onto2MAS Framework, the most important output required from the **Designer** component is ontology model based on end-user requirements. Based on this, the **Generator** component uses `owl` file to generate necessary agents with their behavior, action, resource, rule, and role to instantiate MMDES. Figure 5.3 shows classes of MMDES Ontology.

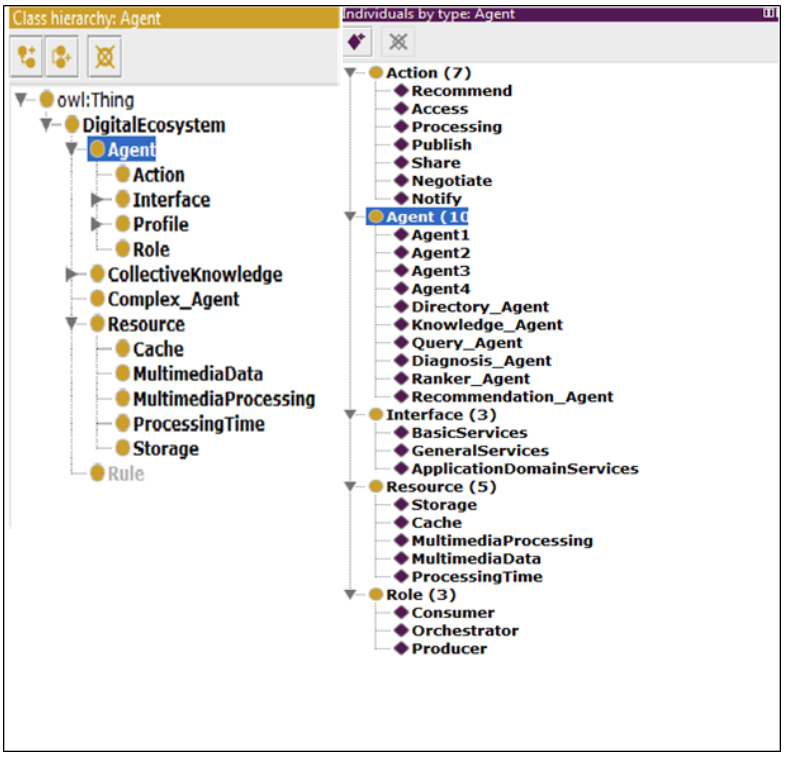

FIGURE 5.3: Classes and Instances of MMDES

5.4 MMDES Agents Generation

Once the design process is over, the **Generator** component retrieves `owl` file to create a set of agents with their related behaviors with respect to their corresponding concepts and rules. Two types of agents are used in MMDES: **General Agents** and **Coordinator Agents**. The scope of these two types of agents are participant and system-wide respectively. Next, we detail the set of agents generated to serve participants and coordinate the whole system.

5.4.1 General Agents

Agents can acquire diverse capabilities to achieve the objectives and meet requirements of MMDES. Besides, these agents can take part in specific duties according to their resources, preferences, and rules in MMDES. Thus, agents are assumed to have various capability. Thus, we present below capabilities that agents should own in MMDES:

- **Identity Management**. Agents should be in charge of keeping and publishing information related to their identity, profile, and preferences. Identity is mainly described by the agent profile and rules and which is important for privacy and security purpose. A profile defines localization, interests, preferences in terms of domains, multimedia contents, applications, storage, etc. Meanwhile, rules are defined according to a participant profile by establishing, for example, conditions to use its contents, storage, and processing capacity, to make exchanges and negotiation, to receive notifications, to keep privacy, to replicate its contents. MMDES ensures that agent rules are respected when another agent uses its resources. This capability enables an agent to the Directory Facilitator (DF) agent to publish their existence, resource availability, and associated rules.

- **Resource Abstraction Service**. This capability provides agents an interface to allow other agents to transparently manage and request access to resources offered. Thus, agents provide a unified way to access different types of resources (enriched texts, images, audio, videos, storage, processing capacities, etc.).

- **Knowledge Management**. An agent stores and manages information related to resources offered to others in the ecosystem. Through this, an agent applies inference rules to enrich and improve the participant knowledge.

- **Query Handling**. An agent is in charge of making queries and requirements of resources available in MMDES. An agent should offer a query interface through which participants can define their queries.

5.4.2 Coordinator Agents in MMDES

In the framework of Onto2MAS, we have shown that coordinator agents are requested to facilitate communication, coordinate, and enforce system level rules. Thus, OnToJade

provided a Moderator Agent, in addition to special agents, Directory Facilitator (DF) and Agent Management System (AMS), from JADE platform. As MMDES is substantial implementation of Digital Ecosystem for multimedia sharing and collaboration, it requires more coordinator agents than Moderator, DF, and AMS. In what follows, we describe the most important special agents to coordinate MMDES in different ways. These agents plays a system-wide roles for the proper governance of MMDES participants.

- **Directory Agent** (DA) is an extension of DF agent of JADE. DA is responsible of providing mechanisms to discover available resources in MMDES, as well as active participants and their roles. When a participant enters to MMDES, it has to register and announce resources to share. When a participant wants to abandon the ecosystem, it has also to announce its retirement. Thus, DA can update the information. Participants can request DA to know about the existence of a particular resource, a type of resource, etc.

- **Diagnosis Agent** (DAA) provides information and metrics regarding participants behavior and performance. This information is important for recommendation purpose. Measures can be, for example, types of resources requested, accessed, offered, and effectively provided; use ratio in term of quantity and time, type of requested contents, latency to respond queries, respond time, average availability, and any other metric considered by MMDES.

- **Ranker Agent** (RA) is in charge of keeping sorted and updated the list of agents, according to their participation and behavior in MMDES. Thus, some activities in the ecosystem (such as assignation, election, and delegation of authorities or determination of the best participant to response to a request) will be made according to these lists. RA is also in charge of keeping the equilibrium of the ecosystem, since it knows how agents are using resources and services in the ecosystem.

- **Recommendation Agent** (RAA) analyzes the metrics measured and generates recommendation to participants, related to contents and behavior. For example, it can recommend new contents according to participant preferences or recommend to free under-used resources.

As we have mentioned above, MMDES consists of general and special agents to achieve its objectives and realize its requirements. To show how MMDES addresses those objectives, we consider knowledge and query management and equilibrium aspects for implementation. In the following sections, we present in detail these two aspects.

5.4.3 Knowledge and Queries Management in MMDES

MMDES manages individual knowledge and generation of collective knowledge (based on relations, behavior, and evolution of the ecosystem) through an ontology, generic rules, Knowledge, Directory, and Query Agents. This allows MMDES to cope with specific needs, to manage the participants and collective knowledge, and to answer queries. In what follows, we describe involving agents for the knowledge and query management in MMDES.

As we shown in the previous section, a participant can have a number of agents to run its interaction and collaboration with others in MMDES. Among the generated agents, Knowledge, Directory, and Query Agents are engaged in the task of knowledge and queries management as described below. These agents are implemented by the Onto2MAS Framework. The specification of each agent is defined by the commands of the OJ language like `SEND` (to model the interactions) and `EXEC` (to execute the specific task) to generate them with their appropriate capabilities and roles.

1) Knowledge Agents (KA). These agents keep the information related to resources offered by participants in MMDES. There is a KA per participant. To manage this, KAs use MMDES Ontology (shown in Figure 5.3) and also its own ontologies describing knowledge in specific domains (e.g., ontology for multimedia contents). Furthermore, any kind of domain ontology can be integrated with MMDES to well equip KA, for instance medical documents. Thus, KAs store the knowledge as RDF documents and define the participant EndPoint to accept and respond queries related to this kind of information. KAs contribute with the compliance of contribution for knowledge.

2) Directory Agent (DA). DA provides a cataloging structure which contains references to all resources and participants in MMDES. DF functionality is efficiently locate resources and services in MMDES. This agent works closely with the DF agent of

JADE platform. Users can request DA to know about the existence of a particular resource. Distributed Hash Tables (DHT) is implemented to access resources efficiently in MMDES. As in any P2P network, a resource is mapped to a unique participating agent, which is responsible for storing and managing that resource. A DHT is a distributed data structure that implements this functionality [155]. Given a target resource or service identifier, the locate operation returns an identifier of the participant (Participant EndPoint or Service PORT) responsible for the target resource or service. Locating objects usually involves a distributed search between a small subset of participants that share resource-allocation (or routing) information. To resolve queries, the DF implements a query decomposer with SPARQL [156]. The need to develop decomposer is to hand complex queries into several small queries that can be executed by one EndPoint as well as strategies to integrate retrieved data. In MMDES, the decomposer is helpful for queries against federations of EndPoints (Participant EndPoints) that stores information about the available EndPoints and the ontologies (AGENT_KNOWLEDGE) used to describe the data accessible through these EndPoints. And it also decomposes queries into sub-queries that can be executed by the selected EndPoints.

3) Query Agents (QA). There exists one QA per participant in MMDES which is used to search resources and services offered by other participants in MMDES. QAs offer an interface through which users can define their queries by, for example, resource attributes (rs={r_{id}, description, characteristics, type, location, owner}). When a query is submitted, QA sends it to the DA, containing the key-value pair, in which 'key' is, for example, one resource attribute and 'value' is the value of this attribute that define the wished search. When a query is submitted, the Query Agent asks to the DA for identifing a specific agent. The QA sends the query to the DA, containing the key-value pair, in which 'key' is, for example, one resource attribute and 'value' is the value of this attribute that define the wished search. If Rule 2 (5.3.1) is matched, DA responds with the location of the best participant (Participant EndPoint or Service PORT) to respond the query, then the QA could access directly, in the remote participant, the required resource or service. For complex queries, QA will work in conjunction with DA to access several EndPoints and integrate all received information.

5.4.4 Ensuring Equilibrium in MMDES

In a Digital Ecosystem, each participant must provide a set of resources to share. And as the ecosystem evolves, it must continue providing resources to support the ecosystem equilibrium. In MMDES, the equilibrium is provided thanks to the processing of Diagnosis Agents and Ranker Agents. These agents are characterized by the component of the Onto2MAS Framework to attain the required capacity to support this task of MMDES. For this, OJ language different operators and commands are involved to later get those agents as intended.

In MMDES, to keep the equilibrium, Ranker Agent monitor the participation profile of each individual regarding to requested, accessed, offered, and indeed supplied services and resources in a period of time. Diagnosis Agents are responsible of keeping this information updated. This information allows Ranker Agent to evaluate the ecosystem sustainability, considering the total offered resources and total requested resources. To keep the balance of the ecosystem, Ranker Agent can, for example, suggest to Recommendation Agents the liberation of resources that are not used or motivate to increment availability of most used resources (by replicas, caching, etc.). In the following subsections, we describe the way how resources are computed and the role of different agents involved in the process of attaining equilibrium.

5.4.4.1 Quantifiable and Sharable Resources

To define the participation profile of an agent in MMDES, several scores are computed so that Diagnosis Agents can measure these resources: (i) Multimedia content; (ii) CPU usage; (iii) GPU usage; (iv) Network traffic; and (v) Space to store data. To store score values, we propose tuples (denoted as $<a, b, ..., c>$) accordingly:

1. **Multimedia Content**: with Obtained and Shared criteria -

   $<Type\_MC, Quantity\_O, Quantity\_S>$.

   A content in which a participant gets from others is referred as Obtained multimedia content ($Quantity\_O$). Shared multimedia content indicates the amount of content provided to other participants in response to obtained contents ($Quantity\_S$). Criteria for a correct equilibrium between shared multimedia data and obtained

multimedia data by a given participant are not responsibility of Diagnosis agent by an agent. But total traffic according to each type of content is quantified in order to be audited. Multimedia content type (Type_MC) corresponds to a tag attached to each shared file as defined by the ecosystem. Total size of content obtained and shared (Quantity_O and Quantity_S respectively) is measured in mega bytes.

2. **GPU usage**: with Obtained, Offered, and Shared criteria -

   $<\{Q, et\}, \{Power\_GPU, [T]\}, pt>$. Three different kinds of measures are set to this resources as processing power can be provided or shared at the same time. Obtained refers to the amount of processing power received from other participants in response to a request. The request for GPU resources depends on the task, the application to be used, and the granularity of the operations to be performed. Then, it is denoted as a quantity Q to be defined by the involved application protocols and an estimated execution time (et). Such quantity can be measured, once consumed, as a ratio between execution time and GPU processing capability, and it should be registered by agents that deliver the resource. GPU processing capability is offered as a tuple of the offered GPU processing power (Power_GPU), and a set of time lapses (T) in which it would be available. The effective use of GPU time provided (pt) and shared is expressed in seconds.

3. **CPU usage**: with Obtained, Offered, and Shared criteria -

   $<\{Q, et\}, \{Power\_CPU, [T]\}, pt>$.

   Analogous to GPU usage, but referring to provided and requested CPU usage.

4. **Dedicated network traffic**: with Offered Limit and Effective Used Traffic criteria -

   $<AB, Data\_trans>$.

   This criteria is mainly measures the limit of the offered bandwidth and the amount effectively used from the available bandwidth. Network traffic as a resource offered by an agent to the ecosystem is important when considering the bandwidth needed by operations as data transfer from other agents to a given one or when considering its postulation to special roles in the ecosystem. Offered bandwidth can be a data transfer segment (AB) quantified in mega bytes per second and total transmitted and received data (Data_trans) measured in mega bytes.

5. **Storage space**: with Requested, Used, Offered, and Provided criteria - $<SS\_R, \{SS\_U, tu\}, SS\_O, \{SS\_P, pt\}>$.

   Different types of measures are set for storage space to show that all the requested resources might be not provided, and even obtained, it might not be properly utilized. To measure these aspects, criteria is set to quantify the request, acquired, and used. Storage space requested to the ecosystem (SS_R) can actually be different from storage space effectively provided and in use (SS_U) because of the possibility of denial of requested resources. This is due to either lack of availability or incompatibility of the rules of the agents involved in the exchange of resources. Data storage in use is registered with a tuple (tu) along with the time during which the storage space has been in use. Offered storage space (SS_O) refers to the exclusive local space reserved to be used by the ecosystem. It does not has a time counter associated because it refers to idle time. In case an individual changes its offer, it must to immediately notify the *Ranker Agent* of MMDES. Finally, the tuple for provided storage space (SS_P) is the local storage space being effectively used by other agents of the ecosystem, and the time (pt) during which the resource has been provided. All values of storage space are expressed in mega bytes.

5.4.4.2 Involved Agents for Ensuring Equilibrium

Three agents are involved in maintaining equilibrium of MMDES: Diagnosis, Ranker, and Recommendation Agents. Recommendation and Ranker Agents depend on the result of Diagnosis Agent. In what follows, we detail the role of each of them in detail.

1) Diagnosis Agent. Diagnosis Agent (DAA) measures and analyzes the participant performance of agents in terms of shared resources and their measured values as detailed in the previous section. DAA keeps updated the participant profile of participation. Profile of participation of a participant, determined by the five quantifiable and shared multimedia resources described above (see Section 5.4.4.1), depends on the correct update of the registered values. To define the participation profile of an participant in MMDES, the measured values according to its consumer or provider roles in terms of Requested Resources, Used Resources, Offered Resources, and Supply Resources must

be considered. To achieve that, methods for reporting new resources being used, offered, or available in a agent are defined. These procedures are started in order to take relevant actions and allow agents report changes. DAA is responsible for ensuring the contribution of each individual for resources.

Resources can be either offered, altered its participation level, or deleted from the pool of resources available in MMDES. They can also be assigned to or released from an agent according to the ecosystem needs. Requested resources must be treated by the corresponding agents in charge of resource allocation, and the Diagnosis Agent must be notified about the assignation whether resources are totally or partially allocated, even rejected. Accordingly, Ranker Agent could influence Recommendation Agent in order to request system resources; or it could judge that a given agent behavior does not meet the balance criteria and command it to release and offer resources in order to keep belonging to MMDES.

Diagnosis Agent viability depends on communication started by Agents with identity (when defining initially offered resources), Knowledge Agents (when transmitting/receiving multimedia data), Resource Abstraction Agents (when validating resource sharing with other agents), and Query Agents (when accepting resources provided by other agents). Also, each agent has to inform when a requested resource is effectively supplied by the ecosystem. Each change on the offered resources by a agent, that is not due to supplied resources, must be immediately reported to the Moderator Agent.

The last responsibility of Diagnosis Agent is to periodically reset its diagnosis structure as the Moderator Agent does. While it is implemented, there will also be a control log of every invocation to the Diagnosis Agent including audits with the Ranker Agent and the automatic scheduled restarts.

2) Recommendation Agent. This agent works with the participation profile and history logs (summaries) registered by the corresponding Diagnosis Agent (of the same participant). For all resources offered by the agent, it analyzes the frequency in which they have been accessed, requested, or used by other agents in MMDES. It is necessary to define metrics to describe that frequency in terms of, for example, time, total amount of resources, balance between offers and demands. Recommendation Agent should manage a pool of metrics to use in different scenarios or in different communities. It helps agents for better contribution and cooperation in sharing resources and supporting each other.

The idea with this analysis is to propose to the agent a better configuration of its resources and maximize its participation in MMDES, based on its needs and priorities. A Digital Ecosystem includes self-organization and dynamism of the system, meaning that adjustments in the participation profile can be influenced by the community. Examples of types of suggestions that Recommendation Agent can make regarding multimedia contents are: liberate low interest contents, access new similar contents, extend sharing scope of high interest contents. Suggestions should increase agents participation and exchange, and also generation of new collective knowledge in MMDES. Nevertheless, it is necessary that multimedia contents be classified by descriptive labels at semantic level, which allows to create maps of related contents. Hence, Recommendation Agents can improve the analysis of use metrics and generate better recommendations and new knowledge.

3) Ranker Agent. This agent can judge the need to increase the offer of limited resources and induce Recommendation Agents to request a readjustment of those resources (e.g., by allowing multimedia duplication) to certain agents. Figure 5.4 shows when Ranker Agent prioritize resources to Recommendation Agents. Recommendation Agent is able to analyze its local viability and transmit the recommendation to the agent Resource Abstraction Agent.

5.5 Deployment of MMDES for a Mobile Platform

We implement and deploy manually the version of for MMDES. This was because the Deployer component proposed in Onto2MAS framework was not implemented in OntoJade prototype. However, this experience gives us valuable hints how to develop the Deployer component in the future. Thus, a Java-based mobile application is used to deploy and access multimedia contents provided by Archivo Nacional de Arte Rupestre (ANAR)[1] that is a non-profit organization responsible for collecting information on the rock manifestation in Venezuela whose information is available to the public. Rupestrian manifestations are old signs that first societies made as an attempt to human communication (e.g., signs recorded in rocks, stones drawings, engravings giant ground). The proposed mobile application allows to access ANAR multimedia databases, in a transparently way for users and ecosystem (they do not need to know the query language).

[1] http://www.anar.org.ve/

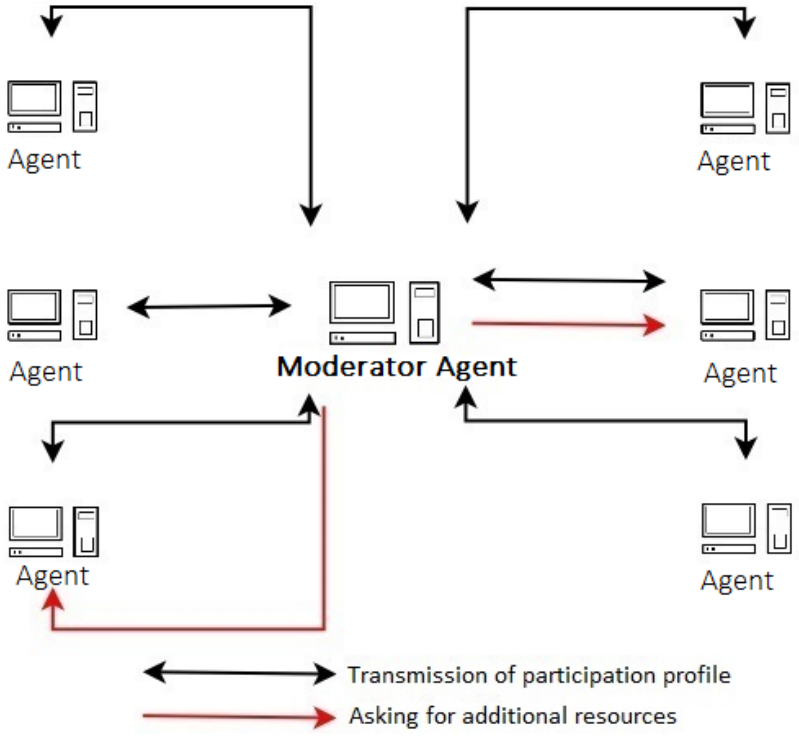

FIGURE 5.4: Ranker Agent prioritize resources to Recommendation Agents

In this implementation, we are limited to the mobile platforms to only offer multimedia contents as resources to share. Moreover, it only can access multimedia contents from other agents even it could technically access other type of MMDES resources. We provide three roles (beginner, consumer, producer) and several profiles (archaeologists, system administrators, students, professors, hikers, general). Since we use Onto2MAS prototype to generate classes with JADE, JADE platform has been adopted for developing distributed applications based on a P2P multi-agent communication architecture

AGENT_RULES and *PROFILES* are expressed using OJ Language of Onto2MAS Framework. To manage privacy of information and resources that users place available in the ecosystem or community, we implemented a P3P protocol [157] to companion Onto2MAS , which is executed by transferring the `owl` files (containing information regarding rules and profile) from the mobile device to corresponding Agents in MMDES, responsible for carrying out the action the individual wants to execute in MMDES (e.g., requests, queries, resource publication). Agent with capability of Identity, Query, and Knowledge Agents were generated to be executed on mobile devices. They represent what a agent wants to offer/access resources to/from other agents under common access

rules. Directory Agent and Knowledge Agent share a small database (developed with SQLite) to keep information related to resources, profile, and rules. This information is available to the Directory Agent in MMDES during the life of the mobile agent. Thus, the Directory Agents can know about resources offered by mobile individuals. Query Agent contains methods for finding multimedia resources considering several parameters related to ANAR contents.

5.5.1 User Registration Process

The registration process of a agent (representing a user) from a mobile device is done through the Moderator and Directory Agents. Moderator Agent knows and controls MMDES generic rules. Along with the registration request, a agent submits its own rules (agent Rules) and profile, with a default role *Beginner*. Thus, the Moderator Agent checks its correctness and its compatibility with MMDES generic rules, before approving or rejecting such registration. If the registration is proceeded, the Moderator Agent informs the Directory Agent about the existence of a new agent in the community along with its new role (*provider/consumer*), offered resources, and scope. Figure 5.5 illustrates the registration process while Figure 5.6 shows some screen-shots of the final application.

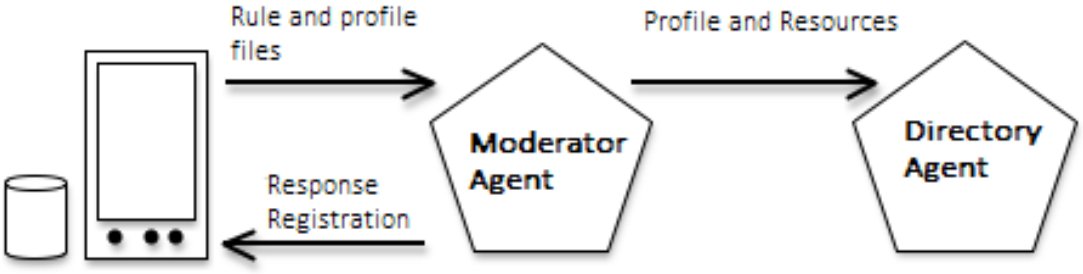

FIGURE 5.5: Registration processes of an Individual from a mobile device

5.5.2 Multimedia Content Queries

To access local data, agent EndPoints use ARQ, a Jena search engine that supports the SPARQL RDF language to do queries[2]. To make queries with ARQ, the query factory

[2]A SPARQL Processor for Jena, The Apache Software Foundation, https://jena.apache.org/documentation/query/

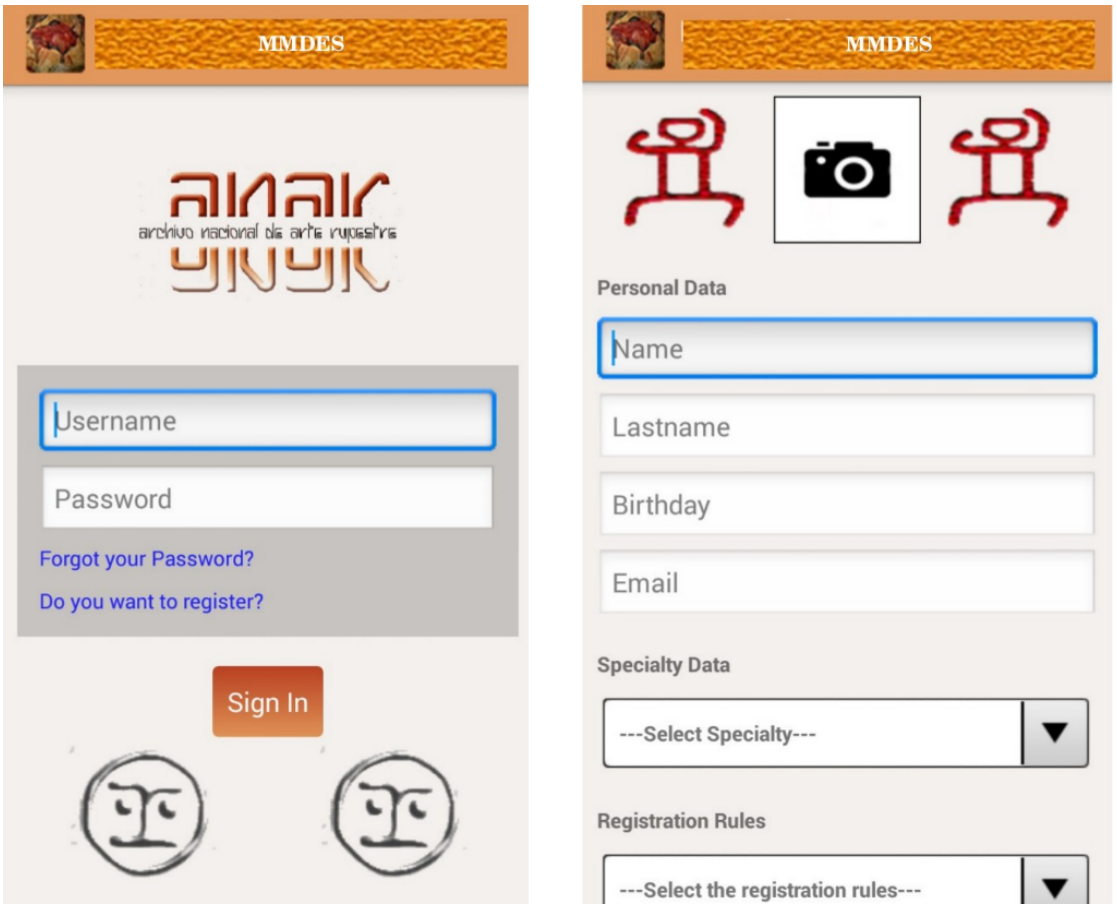

FIGURE 5.6: Registration Process Screenshots

may be created to receive a string representing the query and associates the query with the RDF model, along with parameters needed to make the query. Once the query is executed, a mechanism would be designed to store the result set and then possible results.

Once the agent is registered and is able to access a MMDES community, it can upload any type of multimedia content to share with other agents belonging to the community and define privacy rules for its multimedia resources through the P3P protocol using XML files to restrict or limit *how* and *who* can access them. In our implementation, individuals can share different ANAR multimedia contents (e.g., images, news, types of petroglyphs, photos) and depending on the type of information, it can be accessed according to a particular profile (e.g., archaeologists, system administrators, students, professors, hikers). Thus, some kind of information can be restrictive for general public. Figure 5.8 shows a screen-shot of our retrieval interface.

When a participant makes a query of a resource that is offered in a MMDES community

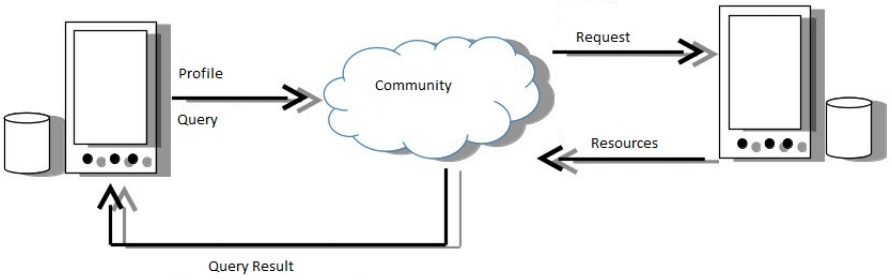

FIGURE 5.7: Query process from a mobile device

(e.g., monoliths in Venezuela), the Query Agent passes the XML file with the user profile (archaeologist, for example) to the Directory Agent. It compares rules of use of each queried resource available in the community and returns as result all those whose rules allow the user profile that performs the search (monoliths photos, videos, georeferencial information, classification documents, etc.). Figure 5.7 graphically shows the query process.

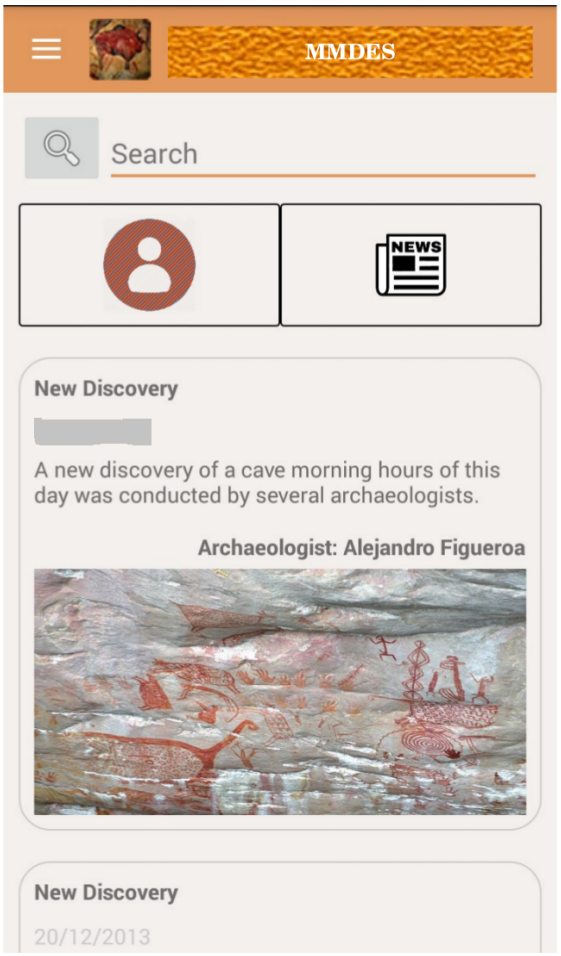

FIGURE 5.8: Retrieval Screenshot

5.6 Conclusion

In this chapter we presented a Multimedia-oriented Digital Ecosystem (MMDES), a special type of Digital Ecosystem which engages in sharing and collaborating multimedia resources among interested participants. This ecosystem is initiated and motivated by the limitations of existing collaborative environments and emerge of Digital Ecosystems. The aim of MMDES is to provide a new collaborative environment that keeps individual interests and benefits of all participants. To achieve this, we employed Onto2MAS Framework that has three steps. The first process focused on designing MMDES depend on the inputs that come from the designer. Thus, MMDES concepts, ontology, and generic rules are provided in detail. The second step is the generation stage, which targeted converting ontological concepts to agents. We identified two types of agents: general and special agents. Coordinator agents are special and core agents that their scope is all through the system whereas general agents are bound to participants. We listed all the agents from both groups for the pertinent implementation of MMDES. To fill this stage of the framework, this work considered mobile devices for deployment. Once MMDES is instantiated, we concentrated on presenting MMDES knowledge and equilibrium management both based on MMDES derived ontology, generic rules, and rightful agents. The last phase is the deployment stage which handles versioning and configuration. Though the deployment of MMDES on mobile devices has done manually, we acquired experience to fully address the deployer component in the future. We detailed the mechanism how the knowledge and query management among the participants. As well, we exhibited the means for measuring the participation and contribution of each participant in keeping the equilibrium of MMDES.

Chapter 6

Conclusion and Future Work

6.1 Discussion

In this thesis work, we mainly account the Digital Ecosystem as one of the approaches to address the design and development of complex systems by mimicking some of the characteristics of biological ecosystems. The work described in the thesis enables to do modeling agents, generate agents, and instantiate Digital Ecosystem of any kind in a simplified and quick way.

Many researchers have been working on Digital Ecosystems in the last decade. They come up with different definitions, concept representations, and models/architectures about this newly introduced collaborative environment. Multi-Agent Systems are also mentioned as a well-suited to modeling systems with heterogeneous, autonomous, and proactive actors. We have summarized the state of the art relevant to the thesis in **Chapter 2**. In this chapter, we first presented definitions, components, characteristics, views, and categories of Digital Ecosystems in order to provide the essential background about Digital Ecosystems. We reviewed the existing Digital Ecosystems to understand the extent to which these diverse systems resemble to biological ecosystems. Following this, we covered related works about frameworks and architectures of Digital Ecosystems and ontology-based MAS methodologies. Then, we investigated agent concept representations in MASs and Digital Ecosystems from seven perspectives and the chapter finalized with a conclusion.

After thoroughly analyzing existing agent models in MASs and Digital Ecosystems in the previous chapter, we propose an ontology model called MAS2DES-Onto. MAS2DES-Onto is described in **Chapter 3**. MAS2DES-Onto provides Agent concept representation that supports the designing and development of MAS-based Digital Ecosystems. It is a generic and comprehensive ontology which considered MAS and Digital Ecosystem environments. MAS2DES-Onto classified the concepts from five views with detail description and their relationships: Structure, Species, Reasoning, Interaction, and System.

In **Chapter 4**, we described Onto2MAS, an ontology-based framework, aimed to support automatic generation of MAS-based Digital Ecosystems. Onto2MAS integrates design, generation, and deployment processes of such complex systems implemented in an easy to use tool called OnToJade. Onto2MAS also provides a language, called OJ, to help the designer in the process of the specification of end-user requirements and rules that will govern the MAS-based Digital Ecosystem. We conducted experimental tests, with OnToJade that demonstrate its efficiency and effectiveness on automatic generation of MAS, without demanding special programming skills. The experimental result assured that the proposed framework is better in handling complexity, less file size, and minimal time for generating a MAS-based Digital Ecosystems. Moreover, the framework is designed to support to any complex systems which could be developed from MAS basis.

Chapter 5 presented a Multimedia-oriented Digital Ecosystem (MMDES) as a special type of Digital Ecosystem for sharing and collaborating multimedia resources among interested participants. This ecosystem is initiated and motivated by the limitations of existing collaborative environments and emerge of Digital Ecosystems to assure individual interests and benefits of all participants. We exercised Onto2MAS Framework and MAS2DES-Onto to design and generate MMDES. This work gave attention to knowledge and query management of individuals in MMDES and implemented a measure for the participation and contribution of individuals in terms of multimedia resources in order to exhibit equilibrium of MMDES and keep the benefits of all participants.

6.2 Contribution

The study presented in this thesis is mainly to deliver a Multimedia Digital Ecosystem with a support of an easy to use framework. In the course of achieving this main objective of this work, we contributed the following:

- **MAS2DES-Onto**. An ontological model, to provide all the essential aspects of agents with their attributes and relationships to support the design and development of MAS-based Digital Ecosystems.
- **Onto2MAS Framework**. Onto2MAS is part of the contribution of this thesis to address the need of framework for Digital Ecosystems development. It is a complete framework for quick and easy development of MAS-based Digital Ecosystems, which integrates MAS2DES-Onto and OJ Language. OJ is a simplified language to help the developer in the process of the specification of end-user requirements.
- **OnToJade Prototype**. It is a first version implementation of Onto2MAS framework. OnToJade consists of different modules for successful implementation of the proposed framework. Through experimental tests, the framework compared with existing approach. The results indicated that our approach simplified the Digital Ecosystem development process.
- **MMDES**. We delivered MMDES as Digital Ecosystem for multimedia sharing and collaboration that ensures the benefits of all its participants. We have shown an implementation of MMDES specifically in terms of addressing individual knowledge management and keeping a balance of the ecosystem.

6.3 Future work

Although we have undertaken extensive works in the way to provide a comprehensive framework for fast and easy development of MAS-based Digital Ecosystems, our efforts offer considerable scope for future works. Some of these are discussed below.

1) Validation of MAS2DES-Onto

We investigated and proposed generic and complete agent concept modeling in the area of MASs and Digital Ecosystems in order to help the development of MAS-based Digital Ecosystems to overcome the problems with existing agent ontologies. We have shown that the proposed framework is exceeding than existing approach. In addition, we have shown the possibilities of using this ontology within our implementations. In this way, the applicability of MAS2DES-Onto has been validated. In the future, we plan to enrich our MAS2DES-Onto by collecting feedback from the research community of the domain. We can conduct a larger empirical test of the usefulness of the MAS2DES-Onto by analysing it and comparing it to other modeling methods.

2) Full implementation of Onto2MAS Framework

We have discussed that the automatic generation of MAS-based Digital Ecosystems with the help of Onto2MAS framework which has three main components. We addressed the first two components of the framework, i.e., **Designer** and **Generator**. However, the third component, **Deployer**, was not fully addressed in this work. The Deployment component is mainly about versioning and configuration of the generated systems for various application. Our experiment and implementation of MMDES has given some experience about the deployment details. Yet, a proper implementation of this component in the framework described in this thesis is still remaining for future work.

3) Enhancing OnToJade Prototype

We have already shown a way of generating a MAS from ontology files in less than a minute time using the OnToJade. In this work, we gave more attention to integrate system level rules. We have also made the necessary rule-based experiments and descriptions at system level, but we would like to investigate the usage of rules at agent level for the purposes of keeping a benefit of a specific agent and other rules that should be addressed.

We have presented our design model for the description of agents and actions that is based on FIPA specifications and JADE platforms. We would like to compare our prototype in other platforms about its performance and efficiency.

4) Full scale Implementation of MMDES

MMDES, as a special type of Digital Ecosystem, provides an appropriate sharing and usage of multimedia resources while keeping individual interests and benefits of all participants. For the moment, we focused on presenting MMDES knowledge and equilibrium management both based on MAS2DES-Onto and Onto2MAS Framework. In addition, it is implemented to provide resources' sharing from mobile device. The next direction is to explore other aspects of Digital Ecosystems to fully implement in MMDES and make it accessible in any type of devices and platforms.

Appendix A

Appendix A: Console outputs

A.1 Multicast Communication

```
Agent AgentA launched !  Got 0 rule(s), 0 variable(s), 1 behaviour(s) and
0 resource(s) path(s) !
Agent AgentB launched !  Got 0 rule(s), 0 variable(s), 2 behaviour(s) and
1 resource(s) path(s) !
Agent AgentC launched !  Got 0 rule(s), 0 variable(s), 2 behaviour(s) and
1 resource(s) path(s) !
Agent AgentD launched !  Got 0 rule(s), 0 variable(s), 2 behaviour(s) and
1 resource(s) path(s) !

Moderator trace:  new agent asks to join the platform, name is:  AgentA
Moderator trace:  new agent asks to join the platform, name is:  AgentB
Moderator trace:  new agent asks to join the platform, name is:  AgentC
Moderator trace:  new agent asks to join the platform, name is:  AgentD

AgentB Trace:  Registered within the DF as a CONTENT-PROVIDER!
AgentC Trace:  Registered within the DF as a CONTENT-PROVIDER!

AgentB Trace:  Sending message(s) to:  Moderator, performative is SUBSCRIBE,
content is C:/Users/Pictures/Anglet.png
AgentC Trace:  Sending message(s) to:  Moderator, performative is SUBSCRIBE,
content is C:/Users/JARS/hello.jar
```

```
AgentA Trace:  Sending message(s) to:  AgentB, performative is REQUEST, content
is null
AgentA Trace:  Sending message(s) to:  AgentC, performative is REQUEST, content
is null

AgentB Trace:  message received from AgentA performative is REQUEST, content
is null
AgentC Trace:  message received from AgentA performative is REQUEST, content
is null

AgentB Trace:  Sending message(s) to:  AgentA, performative is INFORM, content
is C:/Users/Pictures/Anglet.png
AgentC Trace:  Sending message(s) to:  AgentA, performative is INFORM, content
is C:/Users/JARS/hello.jar
AgentA Trace:  received a message from AgentB, performative is INFORM, content
is C:/Users/Pictures/Anglet.png
AgentA Trace:  received a message from AgentC, performative is INFORM, content
is C:/Users/JARS/hello.jar
```

A.2 Broadcast Communication

```
Agent AgentA launched !  Got 0 rule(s), 0 variable(s), 1 behaviour(s) and
0 resource(s) path(s) !
Agent AgentB launched !  Got 0 rule(s), 0 variable(s), 1 behaviour(s) and
0 resource(s) path(s) !
Agent AgentC launched !  Got 0 rule(s), 0 variable(s), 2 behaviour(s) and
1 resource(s) path(s) !
Agent AgentD launched !  Got 0 rule(s), 0 variable(s), 2 behaviour(s) and
1 resource(s) path(s) !_Agent AgentE launched !  Got 0 rule(s), 0 variable(s),
2 behaviour(s) and 0 resource(s) path(s) !
Agent AgentF launched !  Got 0 rule(s), 0 variable(s), 2 behaviour(s) and
0 resource(s) path(s) !
Agent AgentG launched !  Got 0 rule(s), 0 variable(s), 0 behaviour(s) and
0 resource(s) path(s) !
```

```
Moderator trace:  new agent asks to join the platform, name is:  AgentA
Moderator trace:  new agent asks to join the platform, name is:  AgentB
Moderator trace:  new agent asks to join the platform, name is:  AgentC
Moderator trace:  new agent asks to join the platform, name is:  AgentD
Moderator trace:  new agent asks to join the platform, name is:  AgentE
Moderator trace:  new agent asks to join the platform, name is:  AgentF
Moderator trace:  new agent asks to join the platform, name is:  AgentG

AgentC Trace:  Registered within the DF as a CONTENT-PROVIDER!
AgentD Trace:  Registered within the DF as a CONTENT-PROVIDER!
AgentE Trace:  Registered within the DF as a PROCESSING-PROVIDER!
AgentF Trace:  Registered within the DF as a PROCESSING-PROVIDER!

AgentA Trace:  Sending message(s) to:
AgentD,AgentC,AgentB,AgentE,AgentG,AgentF, performative is REQUEST, content
is null
AgentB Trace:  message received from AgentA performative is REQUEST, content
is null
AgentC Trace:  message received from AgentA performative is REQUEST, content
is null
AgentD Trace:  message received from AgentA performative is REQUEST, content
is null
AgentE Trace:  message received from AgentA performative is REQUEST, content
is null
AgentF Trace:  message received from AgentA performative is REQUEST, content
is null
AgentG Trace:  message received from AgentA performative is REQUEST, content
is null

AgentC Trace:  Sending message(s) to:  AgentA, performative is INFORM, content
is C:/Users/Pictures/Anglet.png
AgentD Trace:  Sending message(s) to:  AgentA, performative is INFORM, content
is C:/Users/JARS/hello.jar
AgentA Trace:  received a message from AgentC, performative is INFORM, content
is C:/Users/Pictures/Anglet.png
```

```
AgentA Trace:  received a message from AgentD, performative is INFORM, content
is C:/Users/JARS/hello.jar
```

A.3 System Level Rules

```
Moderator trace:  new agent asks to join the platform, name is:  AgentA
Moderator trace:  new agent asks to join the platform, name is:  AgentB
Moderator trace:  new agent asks to join the platform, name is:  AgentC
Moderator trace:  new agent asks to join the platform, name is:  AgentD

Moderator Trace:  AgentA declaring a new resource !  Name is /screenshot0.png
Moderator Trace:  AgentB declaring a new resource !  Name is /hello.jar
Moderator trace:  Agent B TimeToQuit = 0 !  Name added to kickList !
Moderator Trace:  AgentC declaring a new resource !  Name is /picture.png
Moderator trace:  Agent AgentD have a number of resources inferior to the
constant "contentMinimumAmount", name added to kickList
```

Appendix B

Appendix B: Résumé-Extended

B.1 Contexte

Avec l'évolution rapide de l'Internet et des technologies Web 2.0, il y a eu un changement dans la vision des applications web à l'égard de l'interaction sociale et de collaboration. L'Internet et le Web sont en évolution vers une plate-forme dédiée à la coopération et le partage du contenu créé'epar l'utilisateur. De nos jours, le Web a subit un grand saut d'un espace de producteurs et consommateurs de contenu vers un espace de communautés où chacun peut publier l'information, s'interconnecter, communiquer, collaborer et partager [1]. L'un des aspects dans ce contexte est la croissance importante des réseaux sociaux. Les réseaux sociaux permettent la création et l'échange de contenus générés par les utilisateurs qui se produisent au niveau mondial [2]. Ces réseaux englobent diverses applications axées sur la communication et la collaboration entre les utilisateurs. Le changement de services fournis par une entitéunique en une autre plus complexes intégrants des services multi-intervenants, exige de nouvelles approches pour l'étude de collaboration efficace.

En ce moment, il y a plus de 800 sites de réseaux sociaux actifs et plusieurs plates-formes pour les gens leurs permettant d'interagir. Elles sont caractérisées par la participation, l'ouverture, la connectivitéet le sens de la communauté[3]. Les gens vivent dans un environnement de réseaux sociaux riches, dans lesquels ils peuvent librement et spontanément, générer et partager des contenus de différents types dans le cadre de leurs

activités quotidiennes. Parmi les données partagées sur les réseaux, les données multimédia prennent la plus grande partie. De nombreux médias sociaux ont étécréés pour fournir la possibilitéde partager du contenu multimédia. Des études ont également indiquéque 80 % des informations partagées sur l'Internet sont des données multimédias [4]. Le trafic audio et vidéo combinées devrait atteindre 82 % de tout le trafic Internet en 2018. Par exemple, plus de 500 heures de vidéo sont téléchargées par minute sur YouTube, 8 milliards de vidéos social vus par jour; le site d'hébergement d'images Flickr permet d'accéder à plus de 6 millions de photos. Facebook a plus que 1,7 milliards d'utilisateurs qui téléchargent 300 millions photos par jour. Dernièrement, les messages de format vidéo sont mieux que les autres types de contenu sur toutes les plates-formes car il est partagéprès de quatre fois plus que les autres types. Le partage de vidéo a augmentéde 140 % depuis 2013. Il est même mieux que les photos, que l'on pensait être la meilleure façon d'engager les auditoires sur les réseaux sociaux. En 2015, l'engagement total des utilisateurs des médias sociaux sur le contenu vidéo ont augmentéde 255 % par rapport à 2014. Cela montre que le partage de contenus multimédia est de plus en plus populaire sur Internet. Ainsi, les données multimédia sont capturées, stockées et extraites de différentes sources avec des représentations diversifiées. Pourtant, la technologie de collaboration qui aide les gens à rechercher, utiliser et s'exprimer avec ces médias est en retard [4].

Il existe des systèmes de collaboration tels que les environnements de réseaux sociaux traditionnels (par exemple, client/serveur, peer-to-peer, grid, cloud), qui prennent en charge la communication, la coordination et la coopération [5]. Ce type de système permet à des groupes d'utilisateurs de communiquer et coopérer pour la création, la manipulation et l'accès à une variétéd'informations et de ressources. Leur objectif est de rendre les personnes, les informations et les ressources disponibles pour tous ceux qui ont besoin d'interagir étroitement avec les autres. Néanmoins, ces systèmes de collaboration présentent des restrictions concernant le choix du contenu, la catégorisation, l'agrégation de liens, l'interopérabilité, l'utilisation, le contrôle et la protection de vie privée.

Pour surmonter ces restrictions, les écosystèmes numériques se présentent comme une nouvelle manière de manipuler la collaboration dans un environnement hétérogène et distribué. L'émergence des écosystèmes numériques apporte des avantages substantiels aux participants intéressés permettant la collaboration tout en conservant les avantages mutuels d'en face de systèmes de collaboration traditionnelles qui peuvent seulement

fournir des capacités de collaboration limitées [6]. Les écosystèmes numériques sont souvent décrits comme des systèmes adaptatifs complexes, car ils sont les équivalents numériques des écosystèmes biologiques [7]. Les systèmes Multi-Agents (MAS) sont considérés comme des moyens appropriés pour le développement et la simulation de ces systèmes complexes [8]. Pour l'efficacitéet la qualitéde l'élaboration d'un MAS, un modèle de l'ontologie globale est nécessaire pour représenter les entités et un Framework de développement est nécessaire pour fournir ce système.

Comme nous avons mentionnéci-dessus, de nos jours, les gens sont de plus en plus intéressés à publier et communiquer des contenus multimédias. Ce qui pousse n'importe quel environnement de collaboration proposéà tenir compte de cette réalité. Ainsi, pour surmonter ces limites traditionnelles des environnements de collaboration et pour répondre à l'intérêt croissant dans le partage de ressources multimédia, nous proposons un MMDES axée sur l'écosystème numérique multimédia. MMDES est un type spécial d'Écosystèmes numériques qui met l'accent sur le partage des ressources multimédia (données, méthodes, API/services, capacitéde traitement et de cache et de stockage) entre les participants intéressés.

B.2 Scénario de motivation et défis

Pour motiver notre travail et élaborer les exigences principales adressées par le nouvel environnement collaboratif, nous illustrons le scénario suivant.

Depuis mars 2014, l'Afrique de l'Ouest a connu la plus grande éclosion de fièvre dans l'histoire, avec plusieurs pays touchés (principalement en Guinée, en Sierra Leone, et le Libéria). En réponse à l'épidémie, l'Organisation mondiale de la Santé(OMS) a activéson centre d'opérations d'urgence pour coordonner l'assistance technique et le contrôle des activités avec d'autres partenaires nationaux et internationaux. La campagne de prévention et de contrôle de la tâche a choisi la méthode de l'aide grâce à un documentaire vidéo qui utilise un style visuel pour rassembler différents problèmes de maladies chroniques. Cela aidera à créer une prise de conscience contre l'ignorance de la population sur le virus Ebola. Il est difficile de contrôler cette épidémie. De plus, les gens ont peur et trouvent difficile de croire que la maladie existe. Les communautés n'y sont pas familières. Ainsi, le documentaire pourrait aider la population à mieux comprendre

la maladie et d'obtenir l'appui du reste de la parole dans la lutte contre l'épidémie. En outre, il y a un certain nombre d'organisations locales et internationales engagées dans la prévention et le contrôle du virus. Ces organisations ont leurs propres outils d'analyse et de collecte de données, leurs systèmes et leurs stratégies. Ainsi, il y a eu un appel pour établir des systèmes et des outils qui permettent à l'OMS de répondre rapidement et efficacement dans une stratégie unificatrice. Dans ce contexte, nous décrivons avoir deux scénarios pour illustrer la nécessitéd'un nouvel environnement collaboratif.

Considérons Alice, journaliste à l'OMS, qui a une mission urgente de produire un documentaire portant sur l'épidémie du virus Ebola en Afrique de l'Ouest. L'objectif est de découvrir les schémas de l'éclosion afin de mieux évaluer la propagation du virus en analysant des données à jour sur la crise. En raison d'un risque élevéd'infection et de restrictions de voyage, Alice et son équipe ne peuvent pas filmer dans les régions infectées. Néanmoins, elle s'est rendu compte que de nombreuses personnes et organisations ont fait la publication et le partage de l'information multimédia (c.-à-d., images, clips audio, clips vidéo et textes) à propos de l'épidémie sur tout le web (p. ex., réseaux sociaux) à l'aide de leurs propres moyens et formes. Par conséquent, Alice décide de recueillir les informations disponibles publiquement sur différents milieux sociaux et réseaux liés à l'organisation. Elle va vérifier son intégrité, l'analyser afin de fournir des mises en garde et des pré-études de prédiction. Ensuite, elle va les regrouper afin de créer son documentaire. Mais pour faire cela, Alice s'est rendu compte que les informations sont dispersés et disponibles sur différents réseaux sociaux, de différents natures et formats. Certains sont peu fiables, et non organisés, non ordonnés, abondants et répandus.

Ainsi, afin de produire le film, Alice et son équipe doivent:

- rechercher, sélectionner, et recueillir des données pertinentes (p. ex., films, clips audio, photos) publiées et partagées sur les réseaux sociaux accessibles;
- vérifier l'intégritéet propretédes données (p. ex., éliminer les redondances, éliminer les informations non pertinentes), les classifier et catégoriser (par exemple, par lieu, date, sujet);
- regrouper les données classifiées pour générer des matières à partir de plusieurs sources (p. ex., intégration, fusion, séparation);

- extraire les connaissances à partir de données globales à des fins d'analyse (précautions et études de prévisions);
- utiliser la découverte de connaissances pour regrouper les informations sémantiquement de différents emplacements, services et utilisateurs; et
- produire le film avec un minimum d'effort et d'expertise en programmation et en informatique.

Pour ce faire, Alice est confrontée à plusieurs défis surtout que les plates-formes de collaboration existantes ne peuvent pas résoudre les exigences ci-dessus:

- la sélection et la collecte des contenus appropriés, qui sont difficiles pour la plupart des plates-formes actuelles (p. ex., médias sociaux), afin de soutenir l'édition et le partage de la collecte;
- fournir des plates-formes de contenus qui diffèrent en taille et des sujets qui rendent automatique la classification et la catégorisation compliquée;
- il n'y a aucune description commune technique pour représenter et stocker le contenu;
- il n'y a pas de moyens appropriés pour aider les entités à établir une collaboration appropriée (à l'égard d'un besoin) en fonction de leurs ressources (données et services);
- les plates-formes existantes ne fournissent pas de services appropriés pour plus d'efficacitédans l'utilisation des savoirs collectifs des utilisateurs;
- les utilisateurs ne sont pas en contrôle de leurs données, ce qui entraîne une perte de confiance parmi eux, un non-respect de la vie privée et une restriction de partage et de collaboration des ressources.

Ainsi pour répondre au besoin de l'OMS et en particulier la production de film documentaire d'Alice, il y a un besoin clair et urgent de proposer un nouvel environnement de collaboration qui tient en compte les défis mentionnés. Pour concevoir et développer cet environnement, il nous faut d'abord un modèle de représentation des entités participantes. Suite à cela, il est très important d'avoir une plateforme qui aide à développer facilement et rapidement le système voulu.

B.3 Contribution de la thèse

Le travail présentédans cette thèse est principalement pour offrir un écosystème numérique multimédia avec un appui sur un cadre facile à utiliser. Dans le cadre de la réalisation de cet objectif principal de ce travail, nous contribuons:

- par une base d'Ontologie pour MAS d'écosystème numérique. La première contribution est le développement d'une ontologie générique, globale et basée sur des agents qui vise l'interopérabilité, la communication, l'expression de comportement, les rôles et la définition des règles [9]. Nous fournissons un modèle ontologique, MAS2DES, qui fournit tous les aspects essentiels d'agents avec leurs attributs et relations pour appuyer la conception et l'élaboration d'Écosystèmes numériques MAS indépendamment de tout domaine d'application.

- une plateforme pour le développement d'Écosystèmes numériques de MAS. Pour développer les écosystèmes numériques multimédia pour le partage et la collaboration, nous fournissons une plateforme appelée onto2MAS qui prend en compte de tous les processus de développement d'Écosystèmes numériques de MAS [10]. Il s'agit d'une plateforme complète pour le développement facile et rapide des écosystèmes numériques de MAS, qui intègre MAS2DES-Onto et Langue OJ. OJ est un langage simplifiépour aider le développeur dans le processus de la spécification des besoins de l'utilisateur final.

- un Prototype. Une fois que le modèle de l'ontologie et les plateformes sont livrés, l'étape suivante est le développement d'un prototype. OnToJade est une première version de la mise en œuvre Onto2MAS framework [11]. OnToJade se compose de différents éléments de succès de la mise en œuvre de la plateforme proposée. Grâce à des tests expérimentaux, la plateforme est comparée aux approches actuelles. Les résultats ont indiquéque notre approche simplifie le processus de développement de l'écosystème numérique.

- un Écosystème numérique multimédia pour le partage et la collaboration. L'objectif final de cette thèse est d'offrir un écosystème numérique multimédia pour le partage et la collaboration. Par conséquent, MMDES est livréen tant qu'Ecosystème de gestion et de collaboration, pour des contenus multimédia [9, 12-14]. Ce nouveau

contexte de collaboration porte sur la limitation des médias sociaux et d'autres environnements de collaboration en gardant les avantages mutuels de tous ses participants et d'assurer l'équilibre du système. Nous avons montréune implémentation de MMDES précisément en ce qui concerne la gestion des connaissances individuelles et maintenir un équilibre de l'écosystème. Le développement de ce système prend en compte le projet de la plateforme et de l'ontologie dans ce travail.

B.4 Résumé des chapitres

Dans cette thèse, nous avons principalement considéréde l'écosystème numérique comme l'une des façons d'aborder la conception et le développement de systèmes complexes en imitant certaines caractéristiques des écosystèmes biologiques. Le travail décrit dans cette thèse, permet de concevoir des agents de modélisation, de générer des agents et d'instancier l'écosystème numérique d'une manière simplifiée et rapide.

De nombreux chercheurs ont travaillésur les écosystèmes numériques au cours de la dernière décennie. Ils proposent différentes définitions, représentations, concepts et modèles d'architectures à propos de ce nouveau contexte de collaboration. Les systèmes multi-agents sont également mentionnés comme une bonne adaptation à la modélisation de systèmes autonomes, hétérogènes, et de divulgation d'acteurs. Nous avons résumél'état de l'art pertinent à la thèse dans le Chapitre 2. Dans ce chapitre, nous présentons d'abord les définitions, les composantes, les caractéristiques, les vues et les catégories d'Écosystèmes numériques afin de fournir l'essentiel des écosystèmes numériques à propos de l'arrière-plan. Nous passons en revue les écosystèmes numériques existants pour comprendre la mesure dans laquelle ces divers systèmes ressemblent à des écosystèmes biologiques. Suite à cela, nous couvrons les travaux sur les architectures et les cadres d'Écosystèmes numériques à base d'ontologie et méthodologies. Ensuite, nous étudions les représentations de concept d'agents de l'Écosystèmes numériques à partir de perspectives. Le chapitre final est une conclusion.

Après avoir soigneusement analyséles modèles de l'agent de masse et d'Écosystèmes numériques dans le chapitre 2, le chapitre 3 donne un modèle d'Ontologie pour les écosystèmes numériques de MAS appelé MAS2DES-Onto (montré dans la Figure B.1) . MAS2DES-Onto se compose de tous les aspects essentiels d'agents dans le cadre

d'Écosystèmes numériques de MAS. Le projet de modèle ontologique fournit une représentation du concept de l'agent qui prend en charge la conception et le développement des écosystèmes numériques de MAS. Nous commençons avec les exigences de la modélisation des écosystèmes numériques qui sont lancées depuis les caractéristiques essentielles des deux systèmes. Ensuite, cinq modules de MAS2DES-Onto sont décrits: Structure, espèces, raisonnement, interaction et système. Chaque module se compose de différents concepts définis et d'une relation entre les concepts est clairement indiquée.

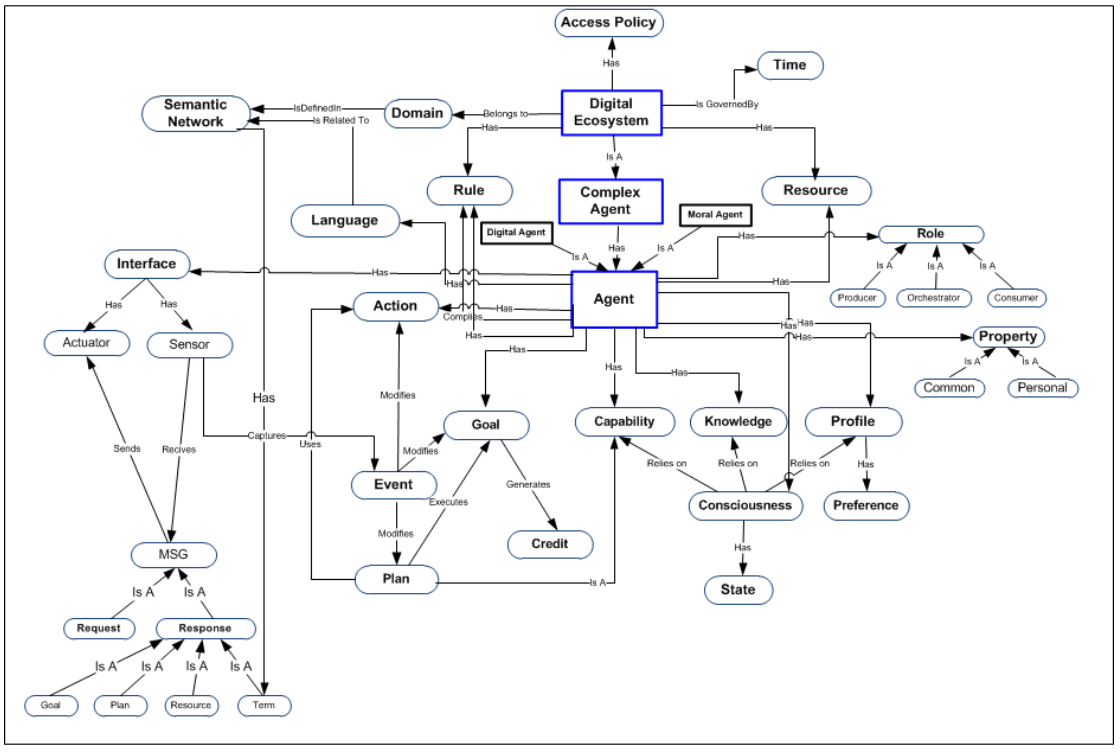

FIGURE B.1: Detailed Conceptual Model for MAS-based Digital Ecosystem

Dans le chapitre 4, nous commençons avec la nécessitéd'une plateforme efficace et de développer des systèmes complexes tels que les écosystèmes. Nous présentons ensuite en détail Onto2MAS Framework pour soutenir la production des écosystèmes basésur MAS avec ses trois composantes principales (montré dans la Figure B.2) . Onto2MAS fournit également un langage, appelée OJ, pour aider le concepteur dans le processus de la spécification des besoins de l'utilisateur final et les règles qui régissent le écosystème numérique basées sur MAS. OnToJade, une implémentation prototype d'Onto2MAS, est également détaillée dans ce chapitre. Les tests expérimentaux et les résultats sont également inclus.

Le chapitre 5 présente un écosystème numérique orientémultimédia (MMDES) comme

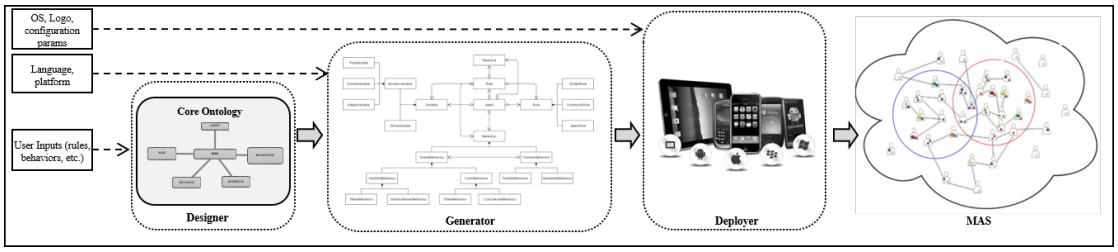

FIGURE B.2: Onto2MAS Framework Components

un type spécial de l'écosystème numérique de partage et de collaboration des ressources multimédia entre les participants. Premièrement, nous décrivons les médias sociaux et les plateformes de collaboration avec leurs limites. MMDES est lancéet motivépar les limitations des environnements collaboratifs et de sortir des écosystèmes numériques pour garantir les intérêts individuels et des avantages de tous les participants. Puis, nous présentons le sommaire des MMDES avec son design et les processus de production. Nous fournissons également le mécanisme de connaissance et de gestion des requêtes et d'assurer l'équilibre de MMDES. Enfin, nous présentons le déploiement de MMDES pour une plate-forme mobile.

Dans le chapitre 6, nous fournissons un résuméde notre travail, et suggérons des orientations futures de la recherche.

B.5 Travaux futurs

Bien que nous ayons entrepris d'importants travaux dans la façon de fournir un cadre global pour le développement facile et rapide des écosystèmes numériques de MAS, nos efforts offrent des possibilités considérables pour nos travaux futures. Certains d'entre eux sont discutés ci-dessous.

1) item La validation de MAS2DES-Onto

Nous avons étudiéet proposéune modélisation complète et générique du concept d'agent dans le domaine de la masse et de l'Écosystèmes numériques afin de favoriser le développement d'Écosystèmes numériques de MAS pour surmonter les problèmes avec les ontologies de l'agent. Nous avons montréque la plateforme proposée est meilleure que les approches existantes. En outre, nous avons montréles possibilités d'utilisation de cette ontologie dans nos implémentations. De cette façon, l'applicabilitéde MAS2DES-Onto a étévalidée.

Dans l'avenir, nous prévoyons d'enrichir notre MAS2DES-Onto en recueillant les commentaires de la communautédes chercheurs du domaine. Nous pouvons effectuer un plus grand essai empirique de l'utilitédu MAS2DES-Onto en l'analysant et en le comparant à d'autres méthodes de modélisation.

2) la mise en œuvre intégrale de Onto2MAS framework

Nous n'avons discutéque la génération automatique des MAS et des écosystèmes avec l'aide d'Onto2MAS framework qui a trois composantes principales. Nous avons abordéles deux premières composantes du cadre, c.-à-d., concepteur et générateur. Cependant, la troisième composante, ≪ le déploiement ≫, n'a pas étépleinement prise en compte dans ce travail. Le déploiement est principalement au sujet de la gestion des versions et de la configuration du produit systèmes pour diverses applications. Notre expérience et la mise en œuvre de MMDES a donnécertaines expériences sur le déploiement de détails. Pourtant, une bonne mise en œuvre de cette composante dans la plateforme décrite dans cette thèse est l'objectif des travaux futurs.

3) Améliorer l'OnToJade Prototype

Nous avons déjà montréune façon de générer un MAS à partir de fichiers d'ontologie en moins d'une minute en utilisant l'OnToJade. Dans ce travail, nous avons donnéplus d'attention à l'intégration au niveau du système de règles. Nous avons également pris les expériences à base de règles et descriptions au niveau du système, mais nous tenons à faire enquête sur l'utilisation de règles au niveau de l'agent aux fins de la tenue d'un avantage d'un agent spécifique et d'autres règles qui devraient être abordées. Nous avons présenténotre modèle de conception pour la description d'agents et d'actions qui est basée sur les spécifications FIPA et jade plates-formes. Nous voudrions comparer notre prototype avec d'autres plates-formes sur son rendement et efficacité.

4) Mise en Œuvre complète de MMDES

MMDES, comme un type spécial d'écosystème numérique, prévoit un partage appropriéet une utilisation de ressources multimédias tout en gardant les intérêts individuels et les avantages de tous les particuliers. Pour le moment, nous avons mis l'accent sur la présentation et la gestion de l'équilibre des connaissances MMDES toutes deux basées sur MAS2DES-onto et onto2MAS Framework. De plus, il est mis en œuvre pour fournir des ressources partagées à partir de l'appareil mobile. La direction suivante est d'explorer

d'autres aspects de l'écosystème numérique à appliquer pleinement dans MMDES afin de le rendre accessible à tout type d'appareils et de plates-formes.

B.6 Conclusion

Nous avons décrit l'utilisation des ontologies dans le développement de MAS et nous avons analyséla proposition MAS2DES-Onto en mettant l'accent sur la conception et le développement d'Écosystèmes numériques de MAS. Nous avons analyséet comparénotre proposition aux œuvres existantes dans la masse des écosystèmes numériques. L'ontologie proposée a montréqu'elle est adaptée pour des MAS et des écosystèmes. Nous avons également discutédes différents modules du MAS2DES-Onto avec du détail sur les concepts et leurs relations. Une plateforme qui permet la traduction de l'ontologie aux codes et modèles de création d'agent est une condition nécessaire pour le développement d'Écosystèmes numériques de MAS. Onto2MAS joue ce rôle d'appui à la génération simple et rapide des systèmes avec d'autres- demande des compétences linguistiques avancés de programmation et sans aucune conscience de la plate-forme sous-jacente. Nous avons illustrél'utilisation de la plateforme pour la génération automatique des MAS et des écosystèmes marins de l'ontologie sous forme de fichiers. Nous avons montréque les capacités de l'infrastructure avec la mise en œuvre de notre prototype. Nous avons fait une comparaison avec l'approche Java/JADE et également différents tests pour démontrer la fonctionnalitéet l'applicabilité du prototype.

Bibliography

[1] Kafui Monu. A conceptual modeling method to use agents in systems analysis. *Proceedings of CAiSE-DC*, page 15, 2008.

[2] Stéphane Faulkner, Manuel Kolp, Yves Wautelet, and Youssef Achbany. *A Formal Description Language for Multi-Agent Architectures*, pages 143–163. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[3] Arash Mousavi, MDJ Nordin, Zulaiha A Othman, et al. An ontology driven, procedural reasoning system-like agent model, for multi-agent based mobile workforce brokering systems. *Journal of Computer Science*, 6(5):557, 2010.

[4] José Alberto RP Sardinha, Ricardo Choren, Viviane Torres da Silva, Ruy Milidiú, and Carlos JP de Lucena. A combined specification language and development framework for agent-based application engineering. *Journal of Systems and Software*, 79(11):1565–1577, 2006.

[5] Giovanni Caire, F Leal, P Chainho, R Evans, F Garijo, JJ Gomez-Sanz, J Pavon, P Kerney, J Stark, and P Massonet. Message: Methodology for engineering systems of software agents. *Technical Information-European Institute for Research and Strategic Studies in Telecommunications*, 2001.

[6] Juan Pavón and Jorge Gómez-Sanz. Agent oriented software engineering with ingenias. In *International Central and Eastern European Conference on Multi-Agent Systems*, pages 394–403. Springer, 2003.

[7] Artur Freitas, Lucas Hilgert, Sabrina Marczak, Felipe Meneguzzi, Rafael H Bordini, and Renata Vieira. A multi-agent systems engineering tool based on ontologies. In *International Conference on Conceptual Modeling*. Springer, 2015.

[8] Hai Dong and Farookh Khadeer Hussain. Digital ecosystem ontology. In *IEEE Conference on Emerging Technologies and Factory Automation (EFTA)*, pages 814–817. IEEE, 2007.

[9] Linda SL Lai and EfraiInt J Spat Data Infrastruct Resm Turban. Groups formation and operations in the web 2.0 environment and social networks. *Group Decision and Negotiation*, 17(5):387–402, 2008.

[10] Andreas M Kaplan and Michael Haenlein. Users of the world, unite! the challenges and opportunities of social media. *Business horizons*, 53(1):59–68, 2010.

[11] Antony Mayfield. What is social media, 2008.

[12] Lexing Xie and Rong Yan. Extracting semantics from multimedia content: challenges and solutions. In *Multimedia Content Analysis*, pages 1–31. Springer, 2008.

[13] Georgia Bafoutsou and Gregoris Mentzas. Review and functional classification of collaborative systems. *International journal of information management*, 22(4): 281–305, 2002.

[14] Geoffrey Fox, Wenjun Wu, Ahmet Uyar, Hasan Bulut, and Shrideep Pallickara. Global multimedia collaboration system. *Concurrency and Computation: Practice and Experience*, 16(5):441–447, 2004.

[15] Gerard Briscoe. Complex adaptive digital ecosystems. In *International Conference on Management of Emergent Digital EcoSystems*, pages 39–46. ACM, 2010.

[16] Philippe De Wilde and Gerard Briscoe. Digital ecosystems: stability of evolving agent populations. In *International Conference on Management of Emergent Digital EcoSystems*, pages 36–43. ACM, 2009.

[17] Solomon Asres Kidanu, Yudith Cardinale, Richard Chbeir, Victor De Ponte, Alejandro Figueroa, Ronier Rodríguez, and Carlos A. Raymundo Ibanez. Mmdes: Multimedia digital ecosystem - new platform for collaboration and sharing. In *International Conference on Computational Science and Engineering*, pages 393–400. IEEE, 2016.

[18] Solomon Asres Kidanu, Richard Chbeir, and Yudith Cardinale. Mas2des-onto: Ontology for mas-based digital ecosystems. In *32nd ACM Symposium on Applied Computing*, Submiitted Paper, 2016.

[19] Solomon Asres Kidanu, Corentin Donzelli, Richard Chbeir, and Yudith Cardinale. Onto2mas: An ontology-based framework for automatic multi-agent system generation. In *International Conference on Signal Image Technology and Internet Based Systems*, page Accepted Paper. IEEE, 2016.

[20] Solomon Asres Kidanu, Yudith Cardinale, Gilbert Tekli, and Richard Chbeir. A multimedia-oriented digital ecosystem: a new collaborative environment. In *International Conference on Computer and Information Science*, pages 411–416. IEEE, 2015.

[21] Minale A Abebe, Fekade Getahun, Solomon Asres, and Richard Chbeir. Event extraction for collective knowledge in multimedia digital ecosystem. In *AFRICON, 2015*, pages 1–5. IEEE, 2015.

[22] Yudith Cardinale, Alejandro Figueroa, Alvaro Parada, Ronier Rodrıguez, Solomon Asres, and Richard Chbeir. EDiM: Ecosistema Digital Multimedia Plataforma Novedosa de Colaboración y Compartimiento. In *Conf. Nacional de Computación, Informática y Sistemas*, pages 14–25, 2015.

[23] Charles B Keating and Polinpapilinho F Katina. Systems of systems engineering: prospects and challenges for the emerging field. *International Journal of System of Systems Engineering*, 2(2-3):234–256, 2011.

[24] John Guckenheimer and Julio M Ottino. Foundations for complex systems research in the physical sciences and engineering. In *Report from an Natural Science Foundation workshop*. Science and Technology Innovation Program of the Woodrow Wilson Internation Center for Scholars, 2008.

[25] Jason Brownlee et al. Complex adaptive systems. *Complex Intelligent Systems Laboratory Technical Report*, pages 1–6, 2007.

[26] Nino Boccara. *Modeling complex systems*. Springer Science & Business Media, 2010. ISBN 978-1-4419-6562-2.

[27] Elizaveta Kuznetsova, Yan-Fu Li, Carlos Ruiz, and Enrico Zio. An integrated framework of agent-based modelling and robust optimization for microgrid energy management. *Applied Energy*, 129:70–88, 2014.

[28] J Stephen Lansing. Complex adaptive systems. *Annual review of anthropology*, pages 183–204, 2003.

[29] Douglas O Norman and Michael L Kuras. Engineering complex systems. In *Complex Engineered Systems*, pages 206–245. Springer, 2006.

[30] Dan DeLaurentis, Robert K Callaway, et al. A system-of-systems perspective for public policy decisions. *Review of Policy Research*, 21(6):829–837, 2004.

[31] R Béjar, MA Latre, J Nogueras-Iso, PR Muro-Medrano, and FJ Zarazaga-Soria. Systems of systems as a conceptual framework for spatial data infrastructures. *International Journal of Spatial Data Infrastructures Research*, 4:201–217, 2009.

[32] Elizabeth Chang and Martin West. Digital ecosystems a next generation of the collaborative environment. In Gabriele Kotsis, David Taniar, Eric Pardede, and Ismail Khalil Ibrahim, editors, *iiWAS*, volume 214, pages 3–24. Austrian Computer Society, 2006. ISBN 3-85403-214-5.

[33] Gerard Briscoe, Suzanne Sadedin, and Greg Paperin. Biology of applied digital ecosystems. In *Inaugural IEEE Digital EcoSystems and Technologies Conference*, pages 458–463. IEEE, 2007.

[34] Madhur Anand, Andrew Gonzalez, Frédéric Guichard, Jurek Kolasa, and Lael Parrott. Ecological systems as complex systems: challenges for an emerging science. *Diversity*, 2(3):395–410, 2010.

[35] PG Balaji and D Srinivasan. An introduction to multi-agent systems. In *Innovations in multi-agent systems and applications-1*, pages 1–27. Springer, 2010.

[36] CJ_ Anumba, OO Ugwu, Leonard Newnham, and A Thorpe. Collaborative design of structures using intelligent agents. *Automation in construction*, 11(1):89–103, 2002.

[37] Melanie Mitchell and Mark Newman. Complex systems theory and evolution. *Encyclopedia of Evolution*, pages 1–5, 2002.

[38] Scott Selberg and Mark A Austin. Toward an evolutionary system of systems architecture. In *International Symposium of The International Council on Systems Engineering*, pages 2394–2047. Citeseer, 2008.

[39] Lars Taxén. A framework for the coordination of complex systems' development. *Dissertation No.800, Linköping University, Dept. of Computer Information Science*, 2003.

[40] Elizabeth Chang and Martin West. Digital ecosystems a next generation of the collaborative environment. In *International Conference on Information Integration and Web-based Applications ServicesS*, pages 3–24. Austrian Computer Society, 2006.

[41] Gerard Briscoe and Philippe De Wilde. Computing of applied digital ecosystems. *International Conference on Management of Emergent Digital EcoSystems*, pages 28–35, 2009.

[42] Jo Stanley and Gerard Briscoe. The abc of digital business ecosystems. *Computing Research Repository*, abs/1005.1899:1–24, 2010.

[43] Harold Boley and Elizabeth Chang. Digital ecosystems: Principles and semantics. *Inaugural IEE-IES Digital Ecosystems and Technologies Conference*, pages 398–403, 2007.

[44] Robert P Biuk-Aghai, Libby Veng-Sam Tang, Simon Fong, and Yain-Whar Si. Wikis as digital ecosystems: An analysis based on authorship. In *International Conference on Digital Ecosystems and Technologies*, pages 581–586. IEEE, 2009.

[45] Mandar Kulkarni and Ron Kreutzer. Building your own digital ecosystem: a holistic approach to enterprise integration. Technical report, Tech. rep., Syntel. http://www. syntelinc. com/uploadedFiles/Syntel_DigitalEcosystem. pdf, 2006.

[46] Francesco Nachira, Andrea Nicolai, Paolo Dini, Marion Le Louarn, and Lorena Rivera Leon. Digital business ecosystems. 2007.

[47] E Chang and M West. Digital ecosystem architecture. pages 268–272, 2006.

[48] Yoosoo Oh, Sébastien Duval, Sehwan Kim, Hyoseok Yoon, Taejin Ha, and Woontack Woo. Foundation of a new digital ecosystem for u-content: Needs, definition, and design. In *International Conference on Virtual and Mixed Reality*, pages 377–386. Springer, 2011.

[49] Peng Liang, Paris Avgeriou, Keqing He, and Lai Xu. From collective knowledge to intelligence: pre-requirements analysis of large and complex systems. In *Proceedings of the 1st Workshop on Web 2.0 for Software Engineering*, pages 26–30. ACM, 2010.

[50] Joanna Saad Sulonen. Digital habitats–stewarding technology for communities. *The Journal of Community Informatics*, 9(3), 2013.

[51] Robert Whelan. Ecological system meets 'digital ecosystem'. In *International Conference on Digital Ecosystems and Technologies*, pages 103–106. IEEE, 2010.

[52] Maja Hadzic and Elizabeth Chang. Application of digital ecosystem design methodology within the health domain. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(4):779–788, 2010.

[53] Gerard Briscoe, Suzanne Sadedin, and Greg Paperin. Biology of applied digital ecosystems. In *Inaugural IEEE Digital EcoSystems and Technologies Conference*, pages 458–463. IEEE, 2007.

[54] John Krogstie. Modeling of digital ecosystems: Challenges and opportunities. In *Working Conference on Virtual Enterprises*, pages 137–145. Springer, 2012.

[55] Cynthia Villalba, Marco Mamei, and Franco Zambonelli. A self-organizing architecture for pervasive ecosystems. In *Self-Organizing Architectures*, pages 275–300. Springer, 2010.

[56] Elizabeth Chang and Martin West. Digital ecosystems a next generation of the collaborative environment. In *iiWAS*, pages 3–24, 2006.

[57] Henrik Sørensen, Dimitrios Raptis, Jesper Kjeldskov, and Mikael B Skov. The 4c framework: principles of interaction in digital ecosystems. In *International Joint Conference on Pervasive and Ubiquitous Computing*, pages 87–97. ACM, 2014.

[58] Paul J Krause, Amir R Razavi, Sotiris Moschoyiannis, and Alexandros Marinos. Stability and complexity in digital ecosystems. In *International Conference on Digital Ecosystems and Technologies*, pages 85–90. IEEE, 2009.

[59] Chen Wu and Elizabeth Chang. Exploring a digital ecosystem conceptual model and its simulation prototype. In *International Symposium on Industrial Electronics*, pages 2933–2938. IEEE, 2007.

[60] Wenbin Li, Youakim Badr, and Frédérique Biennier. Digital ecosystems: challenges and prospects. In *International Conference on Management of Emergent Digital EcoSystems*, pages 117–122. ACM, 2012.

[61] Gerard Briscoe and Philippe De Wilde. Digital ecosystems: evolving service-orientated architectures. In *International conference on Bio inspired models of network, information and computing systems*, page 17. ACM, 2006.

[62] Gerard Briscoe and Philippe De Wilde. Computing of applied digital ecosystems. In *International Conference on Management of Emergent Digital EcoSystems*, pages 5:28–5:35. ACM, 2009.

[63] Francesco Nachira, Paolo Dini, and Andrea Nicolai. A network of digital business ecosystems for europe: roots, processes and perspectives. *Introductory Paper, European Commission*, 2007.

[64] F Nachira. Technologies for digital ecosystems. *European Commission Directorate General Information Society: ICT for Business*, 2004.

[65] Bo Leuf and Ward Cunningham. The wiki way: collaboration and sharing on the internet. 2001.

[66] Jayanta Chatterjee, TV Prabhakar, and Runa Sarkar. Evolution of a digital ecosystem for knowledge services to indian agriculture. *Digital Business Ecosystems Book*, 2007.

[67] Paul P Maglio, Savitha Srinivasan, Jeffrey T Kreulen, and Jim Spohrer. Service systems, service scientists, ssme, and innovation. *Communications of the ACM*, 49(7):81–85, 2006.

[68] Christoph Riedl, Tilo Böhmann, Michael Rosemann, and Helmut Krcmar. Quality management in service ecosystems. *Information Systems and E-Business Management*, 7(2):199–221, 2009.

[69] Yuriko Sawatani. Research in service ecosystems. In *International Conference on Management of Engineering & Technology*, pages 2763–2768. IEEE, 2007.

[70] Thomas Kohlborn, Axel Korthaus, Christoph Riedl, and Helmut Krcmar. Service aggregators in business networks. In *Enterprise Distributed Object Computing Conference Workshops*, pages 195–202. IEEE, 2009.

[71] Georgios Giannakopoulos, Professor Professor Damianos Sakas, Konstadinos Kutsikos, Nikolaos Konstantopoulos, Damianos Sakas, and Yiannis Verginadis. Developing and managing digital service ecosystems: a service science viewpoint. *Journal of Systems and Information Technology*, 16(3):233–248, 2014.

[72] Hai Dong, Farookh Khadeer Hussain, and Elizabeth Chang. A human-centered semantic service platform for the digital ecosystems environment. *World Wide Web*, 13(1-2):75–103, 2010.

[73] Frank-Dieter Dorloff. Service descriptions in digital ecosystems: based on standards and converters. In *International Conference on Digital Ecosystems and Technologies*, pages 75–79. IEEE, 2010.

[74] Pingfeng Liu, Peilu Zhang, and Guihua Nie. Business modeling for service ecosystems. In *International Conference on Management of Emergent Digital EcoSystems*, pages 102–106. ACM, 2010.

[75] César A Marín, Iain Stalker, and Nikolay Mehandjiev. Business ecosystem modelling: Combining natural ecosystems and multi-agent systems. In *International Workshop on Cooperative Information Agents*, pages 181–195. Springer, 2007.

[76] CH Tian, Bonnie K Ray, Juhnyoung Lee, Rongzeng Cao, and Wei Ding. Beam: A framework for business ecosystem analysis and modeling. *IBM Systems Journal*, 47(1):101, 2008.

[77] Pierfranco Ferronato. Architecture for digital ecosystems, beyond service oriented architecture (ieee-dest 2007). In *Inaugural IEEE Digital EcoSystems and Technologies Conference*, pages 660–665. IEEE, 2007.

[78] Jan Sacha, Bartosz Biskupski, Dominik Dahlem, Raymond Cunningham, Jim Dowling, and René Meier. A service-oriented peer-to-peer architecture for a digital ecosystem. In *Inaugural IEEE Digital EcoSystems and Technologies Conference*, pages 205–210. IEEE, 2007.

[79] Gerard Briscoe, Suzanne Sadedin, and Philippe Wilde. Digital ecosystems: Ecosystem-oriented architectures. *Natural Computing: an international journal*, 10(3):1143–1194, 2011.

[80] Gerard Briscoe and Philippe De Wilde. Digital ecosystems: Optimisation by a distributed intelligence. In *International Conference on Digital Ecosystems and Technologies*, pages 192–197. IEEE, 2008.

[81] Eduard Muntaner-Perich and Josep Lluís de la Rosa Esteva. Using dynamic electronic institutions to enable digital business ecosystems. In *Coordination, Organizations, Institutions, and Norms in Agent Systems II*, pages 259–273. Springer, 2007.

[82] Udsanee Pakdeetrakulwong, Pornpit Wongthongtham, Waralak V Siricharoen, and Naveed Khan. An ontology-based multi-agent system for active software engineering ontology. *Mobile Networks and Applications*, 21(1):65–88, 2016.

[83] Vera Maria B Werneck, Luiz Marcio Cysneiros, and Rosa Maria E Moreira Costa. *Modelling Multi-Agent System using Different Methodologies*. INTECH Open Access Publisher, 2011.

[84] Tong ShaoPeng and Zhang Jun. A research on multi agent modeling language. *Procedia Engineering*, 15:1842–1847, 2011.

[85] Viviane Torres Da Silva, Ricardo Choren, and Carlos JP De Lucena. Mas-ml: a multiagent system modelling language. *International Journal of Agent-Oriented Software Engineering*, 2(4):382–421, 2008.

[86] Dariusz Choinski and Michal Senik. Ontology based knowledge management and learning in multi-agent system. In *International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*, pages 65–74. Springer, 2012.

[87] Viviana Mascardi, Daniela Briola, Angela Locoro, Daniele Grignani, Vincenzo Deufemia, Luca Paolino, Nicoletta Bianchi, Henry de Lumley, Damiano Malafronte, and Alessandro Ricciarelli. A holonic multi-agent system for sketch, image and text interpretation in the rock art domain. In *Conference on Agents and Multi-agent Systems- Technologies and Applications*, volume 10, pages 81–100, 2012.

[88] María A Medina, Alfredo Sanchez, and Nohema Castellanos. Ontological agents model based on mas-commonkads methodology. In *International Conference on Electronics, Communications and Computers*, pages 260–263. IEEE, 2004.

[89] Brian Henderson-Sellers. *Agent-oriented methodologies*. IGI Global, 2005.

[90] Scott A DeLoach. The mase methodology. In *Methodologies and software engineering for agent systems*, pages 107–125. Springer, 2004.

[91] Massimo Cossentino. From requirements to code with the passi methodology. *Agent-oriented methodologies*, 3690:79–106, 2005.

[92] Wier Ying, Pradeep Ray, and Lundy Lewis. A methodology for creating ontology-based multi-agent systems with an experiment in financial application development. In *International Conference on System Sciences*, pages 3397–3406. IEEE, 2013.

[93] Fabio Bellifemine, Federico Bergenti, Giovanni Caire, and Agostino Poggi. Jade—a java agent development framework. In *Multi-Agent Programming*, pages 125–147. Springer, 2005.

[94] Janis Grundspenkis and Antons Mislevics. Intelligent agents for business process management systems. *Infonomics for Distributed Business and Decision-Making Environments: Creating Information System Ecology: Creating Information System Ecology*, pages 97–131, 2009.

[95] Alejandro Guerra-Hernández, Amal El Fallah-Seghrouchni, and Henry Soldano. Learning in bdi multi-agent systems. In *International Workshop on Computational Logic in Multi-Agent Systems*, pages 218–233. Springer, 2004.

[96] Stéphane Faulkner, Manuel Kolp, et al. Ontological basis for agent adl. In *International CAiSE Forum on Advanced Information Systems Engineering*, 2003.

[97] Mihaela-Alexandra Puică and Adina-Magda Florea. Emotional belief-desire-intention agent model: Previous work and proposed architecture. *International Journal of Advanced Research in Artificial Intelligence*, 2(2):1–8, 2013.

[98] Ricardo Choren and Carlos Lucena. Modeling multi-agent systems with anote. *Software & Systems Modeling*, 4(2):199–208, 2005.

[99] Csongor I Nyulas, Martin J O'Connor, Samson W Tu, David L Buckeridge, Anna Okhmatovskaia, and Mark A Musen. An ontology-driven framework for deploying jade agent systems. In *International Conference on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 573–577. IEEE, 2008.

[100] Amir Mohamed Talib, Rodziah Atan, Rusli Abdullah, and Masrah Azrafi Azmi Murad. Security ontology driven multi agent system architecture for cloud data storage security:: Ontology development. *International Journal of Computer Science and Network Security*, 12(5):63–72, 2012.

[101] Maja Hadzic, Elizabeth Chang, Tharam Dillon, Janusz Kacprzyk, and Pornpit Wongthongtham. *Ontology-based multi-agent systems*. Springer, 2009.

[102] Jonathan M DiLeo. Ontological engineering and mapping in multiagent systems development. Technical report, DTIC Document, 2002.

[103] Piermarco Burrafato and Massimo Cossentino. Designing a multi-agent solution for a bookstore with the passi methodology. In *Agent Oriented Information Systems*, 2002.

[104] Anupama Mallik, Hiranmay Ghosh, Santanu Chaudhury, and Gaurav Harit. Mowl: An ontology representation language for web-based multimedia applications. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 10(1):8–33, 2013.

[105] Chris van Aart, Ruurd Pels, Giovanni Caire, and Federico Bergenti. Creating and using ontologies in agent communication. In *Proceedings of the Workshop on Ontologies in Agent Systems*, 2002.

[106] Maurizio De Tommasi, Virginia Cisternino, and Angelo Corallo. A rule-based and computation-independent business modelling language for digital business ecosystems. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pages 134–141. Springer, 2005.

[107] Wiebe Van der Hoek and Michael Wooldridge. Multi-agent systems. *Foundations of Artificial Intelligence*, 3:887–928, 2008.

[108] James V Rauff. Multi-agent systems: An introduction to distributed artificial intelligence. *Mathematics and Computer Education*, 39(1):81, 2005.

[109] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009. ISBN 978-0470519462.

[110] Jacques Ferber, Olivier Gutknecht, and Fabien Michel. From agents to organizations: an organizational view of multi-agent systems. In *International Workshop on Agent-Oriented Software Engineering*, pages 214–230. Springer, 2003.

[111] Scott A DeLoach and Jorge L Valenzuela. An agent-environment interaction model. In *International Workshop on Agent-Oriented Software Engineering*, pages 1–18. Springer, 2006.

[112] Ryan Ribeiro de Azevedo, Eric Rommel Galvão Dantas, Fred Freitas, Cleyton Rodrigues, Marcelo J Siqueira C de Almeida, Wendell Campos Veras, and Rubean Santos. An autonomic ontology-based multiagent system for intrusion detection in computing environments. *International Journal for Infonomics (IJI)*, 3(1), 2010.

[113] Chang-Shing Lee and Mei-Hui Wang. Ontology-based computational intelligent multi-agent and its application to cmmi assessment. *Applied Intelligence*, 30(3): 203–219, 2009.

[114] K Yang, AC Lo, and RJ Steele. An ontology-based multi-agent system for the accommodation industry. In *Australian World Wide Web Conference*. IW3C2, 2007.

[115] Maja Hadzic and Elizabeth Chang. Ontology-based multi-agent systems support human disease study and control. *SOAS*, 135:129–141, 2005.

[116] Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa. Developing multi-agent systems with a fipa-compliant agent framework. *Software-Practice and Experience*, 31(2):103–128, 2001.

[117] Holger Knublauch, Ray W Fergerson, Natalya F Noy, and Mark A Musen. The protégé owl plugin: An open development environment for semantic web applications. In *International Semantic Web Conference*, pages 229–243. Springer, 2004.

[118] Daniel M Romero, Wojciech Galuba, Sitaram Asur, and Bernardo A Huberman. Influence and passivity in social media. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 18–33. Springer, 2011.

[119] Rachel Claire Bowley. A comparative case study: Examining the organizational use of social networking sites. 2009.

[120] Yanlin Zheng, Luyi Li, and Fanglin Zheng. Social media support for knowledge management. In *International Conference on Management and Service Science*, 2010.

[121] Sirous Panahi, Jason Watson, and Helen Partridge. Social media and tacit knowledge sharing: developing a conceptual model. *World academy of science, engineering and technology*, (64):1095–1102, 2012.

[122] Kristina Lerman. Social information processing in news aggregation. *Internet Computing*, 11(6):16–28, 2007.

[123] Maeve Duggan, Nicole B Ellison, Cliff Lampe, Amanda Lenhart, and Mary Madden. Social media update 2014. *Pew Research Center*, 9, 2015.

[124] Claudia Canali, Jose Daniel Garcia, and Riccardo Lancellotti. Impact of social networking services on the performance and scalability of web server infrastructures. In *Seventh International Symposium on Network Computing and Applications*, pages 160–167. IEEE, 2008.

[125] Sven Lindmark et al. Web 2.0: Where does europe stand. *Institute for Prospective Technological Studies*, 2009.

[126] Per Andersen. *What is Web 2.0?: ideas, technologies and implications for education*, volume 1. JISC Bristol, 2007.

[127] Robert Wollan, Nick Smith, and Catherine Zhou. *The social media management handbook: Everything you need to know to get social media working in your business.* John Wiley & Sons, 2010.

[128] S Mavromoustakos and K Papanikolaou. E-learning engineering in the web 2.0 era. In *2nd International Conference on Education Technology and Computer*, pages 534–538, 2010.

[129] Michael Eraut. Non-formal learning and tacit knowledge in professional work. *British journal of educational psychology*, 70(1):113–136, 2000.

[130] Karlo Berket, Abdelilah Essiari, and Mary R. Thompson. Securing resources in collaborative environments: A peer-to-peer approach. In *International Congress on Parallel and Distributed Computing and Systems*, pages 588–593. IASTED, 2005.

[131] José Van Dijck and Thomas Poell. Understanding social media logic. *Media and Communication*, 1(1):2–14, 2013.

[132] Abdelhamid Malki, Sidi Mohamed Benslimane, et al. Building semantic mashup. In *International Conference on Web and Information Technologies*, volume 867, pages 40–49. CEUR-WS.org, 2012.

[133] Giusy Di Lorenzo, Hakim Hacid, Hye-young Paik, and Boualem Benatallah. Data integration in mashups. *ACM Sigmod Record*, 38(1):59–66, 2009.

[134] CSA Nextmedia. Social networks overview: Current trends and research challenges. *European Commission Information Society and Media*, 2010.

[135] Sebastian Tramp, Philipp Frischmuth, Timofey Ermilov, Saeedeh Shekarpour, and Sören Auer. An architecture of a distributed semantic social network. *Semantic Web*, 5(1):77–95, 2014.

[136] Rumi Ghosh and Sitaram Asur. Mining information from heterogeneous sources: A topic modeling approach. In *the 19th ACM SIGKDD Workshop on Mining Data Semantics*, 2013.

[137] Min Weng. A multimedia social-networking community for mobile devices. *Interactive Telecommunications Program*, 2007.

[138] Tom Gruber. Collective knowledge systems: Where the social web meets the semantic web. *Web semantics: science, services and agents on the World Wide Web*, 6(1):4–13, 2008.

[139] Gilbert Tekli, Richard Chbeir, and Jacques Fayolle. A visual programming language for xml manipulation. *Journal of Visual Languages & Computing*, 24(2): 110–135, 2013.

[140] John Soldatos, Moez Draief, Craig Macdonald, and Iadh Ounis. Multimedia search over integrated social and sensor networks. In *International conference companion on World Wide Web*, pages 283–286. ACM, 2012.

[141] Matthias Bender, Tom Crecelius, Mouna Kacimi, Sebastian Michel, Josiane Xavier Parreira, and Gerhard Weikum. Peer-to-peer information search: Semantic, social, or spiritual? *IEEE Data Eng. Bull.*, 30(2):51–60, 2007.

[142] M. Bonifacio, R. Cuel, G. Mameli, and M. Nori. A peer-to-peer architecture for distributed knowledge management. In *International Symposium on Multi-Agent Systems, Large Complex Systems, and E-Businesses*, pages 8–10, 2002.

[143] H. M. N. Dilum Bandara and Anura P. Jayasumana. Collaborative applications over peer-to-peer systems-challenges and solutions. *Peer-to-Peer Networking and Applications*, 6(3):257–276, 2013.

[144] Gianpaolo Cugola and Gian Pietro Picco. Peer-to-peer for collaborative applications. In *International Conference on Distributed Computing Systems*, 2002.

[145] Michal Feldman and John Chuang. Overcoming free-riding behavior in peer-to-peer systems. *SIGecom Exch.*, 5(4):41–50, July 2005.

[146] Alek Opitz, Hartmut Konig, and Sebastian Szamlewska. What does grid computing cost? *Journal of Grid Computing*, 6(4):385–397, 2008.

[147] Heinz Stockinger. Defining the grid: A snapshot on the current view. *Jornal of Super computing*, 42(1):3–17, 2007. ISSN 0920-8542.

[148] N.Sandeep Chaitanya, S. Ramachandram, B. Padmavathi, S.Shiva Skandha, and G.Ravi Kumar. Data privacy for grid systems. In *Advances in Computing and Communications*, volume 193, pages 70–78. 2011. ISBN 978-3-642-22725-7.

[149] SheikhMahbub Habib, Sascha Hauke, Sebastian Ries, and Max Muhlhauser. Trust as a facilitator in cloud computing: a survey. *Journal of Cloud Computing*, 1(1): 1–18, 2012.

[150] C.N. Hofer and G. Karagiannis. Cloud computing services: taxonomy and comparison. *Journal of Internet Services and Applications*, 2(2):81–94, 2011. ISSN 1867-4828.

[151] Shouhuai Xu and Moti Yung. Socialclouds: Concept, security architecture and some mechanisms. In *Trusted Systems*, volume 6163 of *LNCS*, pages 104–128. 2010.

[152] Tomomi Kawashima and Jianhua Ma. Tomscop-a synchronous p2p collaboration platform over jxta. In *24th International Conference on Distributed Computing Systems*, pages 85–90. IEEE, 2004.

[153] H. Boley and E. Chang. Digital ecosystems: Principles and semantics. In *Inaugural IEEE Digital EcoSystems and Technologies Conference*, pages 398–403. IEEE, 2007.

[154] Yoosoo Oh, Sébastien Duval, Sehwan Kim, Hyoseok Yoon, Taejin Ha, and Woontack Woo. Foundation of a new digital ecosystem for u-content: Needs, definition, and design. In *International Conference on Virtual and Mixed Reality: Systems and Applications*, pages 377–386, 2011.

[155] Matthew Harren, Joseph M Hellerstein, Ryan Huebsch, Boon Thau Loo, Scott Shenker, and Ion Stoica. Complex queries in dht-based peer-to-peer networks. In *International Workshop on Peer-to-Peer Systems*, pages 242–250. Springer, 2002.

[156] Gabriela Montoya, Maria-Esther Vidal, and Maribel Acosta. Defender: a decomposer for queries against federations of endpoints. In *Extended Semantic Web Conference*, pages 480–484. Springer, 2012.

[157] W3C. Platform for privacy preferences (p3p), 2015. URL http://www.w3.org/P3P/.