UNIVERSITY OF GENOVA

PHD PROGRAM IN BIOENGINEERING AND ROBOTICS

A knowledge-based approach towards human activity recognition in smart environments

by

Syed Yusha Kareem

Thesis submitted for the degree of *Doctor of Philosophy* (33° cycle)

July 2021

Prof. Fulvio Mastrogiovanni — Supervisor

Prof. Giorgio Cannata — Head of the PhD program

Dibris

Department of Informatics, Bioengineering, Robotics and Systems Engineering

I would like to dedicate this thesis to my family, mentors and friends.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Syed Yusha Kareem

July 2021

Acknowledgements

And I would like to thank my supervisor Fulvio Mastrogiovanni: he always gave me the freedom to evolve and make my own decisions. His guidance and support were essential for this work. Special thanks to Luca Buoncompagni and Tommaso Elia: collaborating and discussing with you on challenging problems was always great! I would also like to thank my thesis reviewers Prof. Federico Pecora and Prof. Silvia Rossi for patiently reviewing my thesis and providing valuable feedback. Thanks to my colleagues and friends at TheEngineRoom - Alessandro Carfì, Antony Thomas, Hossein Karami, Kourosh Darvish - for all our jovial discussions and meals together during the past three years. I also want to thank all my remaining friends who I came across and spent time with during my studies in different parts of the world - for all the wonderful experiences I have had with them. Thanks also to my parents and family for always being my sturdy backbone. Finally, above all, I thank The One to whom belongs all praise.

Abstract

For many years it is known that the population of older persons is on the rise. A recent report estimates that globally, the share of the population aged 65 years or over is expected to increase from 9.3 percent in 2020 to around 16.0 percent in 2050 [1]. This point has been one of the main sources of motivation for active research in the domain of human activity recognition in smart-homes. The ability to perform ADL without assistance from other people can be considered as a reference for the estimation of the independent living level of the older person. Conventionally, this has been assessed by health-care domain experts via a qualitative evaluation of the ADL. Since this evaluation is qualitative, it can vary based on the person being monitored and the caregiver's experience. A significant amount of research work is implicitly or explicitly aimed at augmenting the health-care domain expert's qualitative evaluation with quantitative data or knowledge obtained from HAR. From a medical perspective, there is a lack of evidence about the technology readiness level of smart home architectures supporting older persons by recognizing ADL [2]. We hypothesize that this may be due to a lack of effective collaboration between smart-home researchers/developers and health-care domain experts, especially when considering HAR. We foresee an increase in HAR systems being developed in close collaboration with caregivers and geriatricians to support their qualitative evaluation of ADL with explainable quantitative outcomes of the HAR systems. This has been a motivation for the work in this thesis. The recognition of human activities – in particular ADL – may not only be limited to support the health and well-being of older people. It can be relevant to home users in general. For instance, HAR could support digital assistants or companion robots to provide contextually relevant and proactive support to the home users, whether young adults or old. This has also been a motivation for the work in this thesis.

Given our motivations, namely, (i) facilitation of iterative development and ease in collaboration between HAR system researchers/developers and health-care domain experts in ADL, and (ii) robust HAR that can support digital assistants or companion robots. There is a need for the development of a HAR framework that at its core is modular and flexible to facilitate an iterative development process [3], which is an integral part of collaborative work that

involves develop-test-improve phases. At the same time, the framework should be intelligible for the sake of enriched collaboration with health-care domain experts. Furthermore, it should be scalable, online, and accurate for having robust HAR, which can enable many smart-home applications. The goal of this thesis is to design and evaluate such a framework.

This thesis contributes to the domain of HAR in smart-homes. Particularly the contribution can be divided into three parts. The first contribution is Arianna+, a framework to develop networks of ontologies - for knowledge representation and reasoning - that enables smart homes to perform human activity recognition online. The second contribution is OWLOOP, an API that supports the development of HAR system architectures based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object-Oriented Programming (OOP). The third contribution is the evaluation and exploitation of Arianna+ using OWLOOP API. The exploitation of Arianna+ using OWLOOP API has resulted in four HAR system implementations. The evaluations and results of these HAR systems emphasize the novelty of Arianna+.

Table of contents

List of figures **ix**

List of tables **xii**

List of abbreviations **xiii**

1 Introduction **1**

1.1 Human Activity Recognition (HAR) 1

1.1.1 Motivation 2

1.1.2 Challenges and problem description 3

1.2 Contributions 4

1.2.1 First contribution 4

1.2.2 Second contribution 5

1.2.3 Third contribution 6

1.3 A brief guide to reading this thesis 9

2 Related work **11**

2.1 HAR for health and well-being in smart homes 11

2.1.1 Activities of Daily Living (ADL) 13

2.1.2 Ambient Assisted Living (AAL) 14

2.2 Sensing approaches for HAR 15

2.2.1 Visual sensor-based 15

2.2.2 Non-visual sensor-based 16

2.3 Modeling approaches for HAR 18

2.3.1 Data-driven 18

2.3.2 Knowledge-driven 19

2.3.3 Hybrid 21

2.4 Research problems addressed by this thesis 23

3 Arianna+ **27**
3.1 Preliminaries . . . 27
3.1.1 Description Logic (DL) and Ontology Web Language (OWL) . . . 27
3.2 Arianna+ . . . 28
3.2.1 Novelty . . . 28
3.2.2 Description . . . 29
3.3 Core components . . . 31
3.3.1 Statements . . . 31
3.3.2 Fluent models . . . 32
3.3.3 Computational procedures . . . 35
3.3.4 Contextualized reasoning using multiple ontologies . . . 36
3.4 Summary . . . 40

4 OWLOOP API **42**
4.1 Preliminaries . . . 42
4.1.1 Background . . . 42
4.2 OWLOOP API . . . 44
4.2.1 Novelty . . . 44
4.2.2 Description . . . 44
4.3 Core components . . . 46
4.3.1 Definition of a Descriptor . . . 46
4.3.2 Construction of Descriptors . . . 49
4.3.3 Usage of Descriptors . . . 50
4.4 Summary . . . 55

5 Evaluating and exploiting Arianna+ using OWLOOP API **57**
5.1 Evaluating computational performance of a network of ontologies . . . 58
5.1.1 Introduction . . . 58
5.1.2 Methodology . . . 58
5.1.3 Experimental evaluation . . . 62
5.1.4 Summary . . . 68
5.2 A HAR system based on Arianna+ evaluated using the CASAS dataset . . . 69
5.2.1 Introduction . . . 69
5.2.2 Methodology . . . 70
5.2.3 Experimental evaluation . . . 71
5.2.4 Summary . . . 90

5.3 A real-world HAR system showcasing explainable results using SPARQL queries . . . 92
5.3.1 Introduction . . . 92
5.3.2 Methodology . . . 93
5.3.3 Experimental evaluation . . . 94
5.3.4 Summary . . . 99
5.4 Online HAR enables a smart-home application with a companion robot . . . 103
5.4.1 Introduction . . . 103
5.4.2 Methodology . . . 104
5.4.3 Experimental evaluation . . . 109
5.4.4 Summary . . . 118

6 Conclusion and future work 119
6.1 Conclusion . . . 119
6.2 Future work . . . 124

References 128

Appendix A Representing modules of complex HAR system architectures as agents and analyzing them from the lens of network science 142

List of figures

3.1 Graphical representation of the fluent model of activity $\widetilde{A}_1$. 33
3.2 Graphical representation of the fluent model of activity $\widetilde{A}_7$. 34

4.1 A simple ontology used as an example throughout Chapter 4. 46
4.2 The UML notations used in Chapter 4. 48
4.3 The general-purpose definition of OWLOOP `Descriptor`. Colours identify packages, *i.e.*, related classes, and they are also used in Figures 4.4–4.6. . . 49
4.4 The implementation of the `FullClassDescriptor`. 50
4.5 The implementation of the `FullIndividualDescriptor`. 54
4.6 The implementation of the `FullObjectPropertyDescriptor`. 55

5.1 A simplified ontology network $\mathcal{O}$. 60
5.2 Visual representation of statements that make up the $\mathcal{A}_2$ model: statements are shown as vertical arrows where dashed arrows indicate information from $\mathcal{P}$, and solid arrows indicate statements generated by this model. Statement indexes indicate sensors influencing the state of that statement, while the temporal restrictions are shown as black lines. 60
5.3 The system's architecture for the work presented in Section 5.1. Where, the link *rsd* signifies the flow of raw sensor data, *asds* signifies the flow of aggregated sensor data in the form of statements, *ias* signifies inferred activity statements and f signifies frequency. 62
5.4 The simulated dataset used with the HAR system presented in Section 5.1. . 65
5.5 Ontology network's complexity *versus* reasoner's computation time. On x-axis are the number of ontologies, where 2 means $(\mathcal{P} + \mathcal{A}_1)$, 3 means $(\mathcal{P} + \mathcal{A}_1 + \mathcal{A}_2)$, etc. 66

5.6 The sensors of a smart home, *i.e.*, motion detectors M_i, item-presence sensors I_i, flow sensors F_i, door sensors D_i, and phone P_i. Sensors are contextualized in a topological map concerning areas and furniture. 71
5.7 This figure partially shows the hierarchies of classes and sub-classes in the ontologies. They represent the prior knowledge used to address the scenario presented in Section 5.2.3. 73
5.8 On the right hand side is the ontology network for the HAR system presented in Section 5.2 and developed based on Arianna+ for recognizing activities in the CASAS dataset. This network is bootstrapped from the upper ontology shown on the left hand side. 82
5.9 A graphical representation of the fluent models to classify the activities in the CASAS dataset. 82
5.10 The following figures show behaviour of the system. Graph (a) shows the experiment and the associated activity recognitions. Graphs (b) and (c) show the reasoning performance. Graph (d) shows the events in the network and the statements X propagated from the spatial ontology $\mathscr{L}$ to the relevant activity ontology $\mathscr{T}_a$. 86
5.11 The spatial ontology $\mathscr{L}$ reasoning time against ontology complexity. 88
5.12 Volunteer performing an experiment at the smart-home. Left image shows volunteer in the Bathroom. Right image shows volunteer in the Kitchen. . . 94
5.13 The system's architecture for the work presented in Section 5.3. 95
5.14 Kitchen ontology shown in Protégé editor. 96
5.15 Query ontology shown in Protégé editor. 96
5.16 Visual representation if the `HavingBreakfast` activity model, i.e., $\widetilde{A}_1$. . . . 97
5.17 SPARQL `Query 1` and its result. 99
5.18 SPARQL `Query 2` and its result. 100
5.19 SPARQL `Query 3` and its result. 100
5.20 SPARQL `Query 4` and its result. 101
5.21 SPARQL `Query 5` and its result. 101
5.22 SPARQL `Query 6` and its result. 102
5.23 The system's architecture for the work presented in Section 5.4. 104
5.24 Ontology as seen in the Protégé editor. 107
5.25 Figure on the left shows robot assistant (Sota) and figure on the right shows vocal assistant. 108

5.26 Different moments of the video recording from the ego-centric point of view - for medication reminder scenario. 110
5.27 Example of the responses to the items of the scale Perspicuity. 112
5.28 Vocal assistant six scales evaluation 113
5.29 Histogram shows evaluation (mean and variance on the six scales) of the system equipped with Robot assistant interface. 115
5.30 Comparison between the interaction with robot assistant and vocal assistant 116

A.1 An example of a complex HAR system's architecture. 143
A.2 A network developed using Gephi, where nodes represent agents and links represent communication between the agents. The color of nodes represents different communities and the color of links represents the type of communication between them. 144

List of tables

1.1 Human Activity Recognition Models in Ontology Networks. 5
1.2 OWLOOP: A modular API to describe OWL axioms in OOP objects hierarchies. 6
1.3 Arianna +: Scalable Human Activity Recognition by reasoning with a network of Ontologies. 7
1.4 A smart home system with a digital assistant - Comparison between a vocal-assistant and a robot-assistant. 8

4.1 Code metadata of OWLOOP API. 45
4.2 The definition of OWL and OWLOOP entities. The cardinality C concerns an OWL axiom among $\{\mathsf{some}, \mathsf{only}, \mathsf{min}\,n, \mathsf{max}\,n, \mathsf{exact}\,n\}$ where $n \in \mathbb{N}^{+}$. . 47
4.3 The expressions which the current version of OWLOOP can map and their structure in descriptors. 48

5.1 The design of the interface among computational procedures and ontologies for the network shown in Figure 5.8. 74
5.2 Confusion matrix of the activity recognition rate obtained with the ontology network. 89
5.3 F-measure of activity recognitions shown for the ontology network developed with Arianna+ and approaches using a single context to represent data. . . . 90
5.4 Evaluation (mean and variance) of the system equipped with Vocal assistant. 113
5.5 Vocal assistant interface compared to benchmark. 114
5.6 Evaluation (mean and variance) of the system equipped with Robot assistant. 114
5.7 Robot assistant interface compared to benchmark.. 115
5.8 Comparison between the interaction with Vocal assistant and Robot assistant. 116
5.9 T-Test comparison between the interaction with Vocal assistant and Robot assistant. 117

List of abbreviations

HAR	Human Activity Recognition
ADL	Activities of Daily Living
API	Application Programming Interface
OWL	Ontology Web Language
SPARQL	SPARQL protocol and Query Language
AAL	Ambient Assisted Living
BADL	Basic Activities of Daily Living
IADL	Instrumental Activities of Daily Living
RFID	Radio-Frequency Identification
DL	Description Logic
W3C	World Wide Web Consortium
MLN	Markov Logic Networks
RQ	Research Question
SWRL	Semantic Web Rule Language
OOP	Object-Oriented Programming
OWLOOP	Ontology Web Language Object Oriented Programming
HCI	Human Computer Interaction
ML	Machine Learning

RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
IRI	Internationalized Resource Identifier
MAS	Multi Agent System

Chapter 1

Introduction

1.1 Human Activity Recognition (HAR)

Human activity recognition (HAR) has been a very active research topic in the past two decades for its applications in various fields such as health care, sports, security and surveillance, and human-computer interaction. Activity recognition can be defined as the ability to recognize/detect current activity based on of information received from different sensors [4]. Activities can be categorized at multiple granularity levels. In [5], to define human activities at different complexity levels and durations, the authors distinguish Activities of Daily Living (ADL) from actions - in a smart home. An action is an atomic activity that is performed by a single subject and lasts for a relatively short time. Some examples of action are "pouring" and "drinking". An ADL is usually defined as a more complex activity, which lasts for a longer time than an action. Often ADL consist of an orderly succession of simpler activities. For instance, "Cleaning home" may consist of: "Taking cleaning tools", "Walking", and "Putting back cleaning tools". The ordering of simple activities may depend on an individual's preferences or habits, thus leading to several variants of an activity. Furthermore, activities may have time-related connections to each other, to form a composite activity. With respect to this, we may distinguish three situations: sequential activities, concurrent activities, and interleaved activities [6].

Human activity recognition (HAR) is a highly dynamic and challenging research topic. It aims at determining the activities of a person or a group of persons based on visual and non-visual sensory data [7], as well as, based on knowledge about the context (environmental, spatial, temporal, etc.) within which the observed activities take place. These sensors can be cameras, wearable devices, sensors attached to objects of daily use, or sensors deployed in the environment. With the advancements in technology and the reduction in device costs,

particularly of smartphones and smartwatches, the logging of daily activities has also become very popular and practical. People are logging their daily life activities, such as drinking water, cooking, eating, sleeping, walking, etc. Nonetheless, challenges still exist in order to recognize these activities automatically.

Within the domain of smart-home environments, researchers have developed and investigated HAR systems with various applications in mind. We divide them roughly into four categories: namely, health-oriented [8], security-oriented [9], comfort/safety-oriented [10, 11], and energy-oriented [12]. Health and well-being-oriented systems are those that monitor the status of the user (e.g. weight, heart-rate, ADL, etc.). Security-oriented systems detect distress or hazardous situations, for instance, smoke detection, intrusion detection, etc. Comfort-oriented systems monitor user activities and context to provide classical home automation that allows them to manage home appliances easily, and safety oriented systems detect emergencies such as detecting falls. Finally, energy-oriented systems analyze electricity consumption to approximately recognize user activities and make recommendations such that the home user can save electricity and costs.

1.1.1 Motivation

It is estimated that there will be an increase in the size of the older population between 2020 and 2050. Globally, the share of the population aged 65 years or over is expected to increase from 9.3 percent in 2020 to around 16.0 percent in 2050 [1]. The same report highlights that for older persons to live independently, some universal needs must be met, which include health care services. This particular point is not new and has been relevant for many years. Furthermore, it has been one of the motivations for smart home research - particularly in the past decade - due to advances in sensing, networking, and ambient intelligence technologies.

To support healthy and active aging, to maintain and improve the quality of life of older persons, and to respond to the needs of the rapidly aging population – researchers are developing smart-home environments that can recognize Activities of Daily Living (ADL). The ability to perform ADL without assistance from other people can be considered as a reference for the estimation of the independent living level of the older person [2]. Conventionally, this has been assessed by the caregivers and geriatricians – i.e., the domain experts – via a qualitative evaluation of the ADL [13]. ADL are usually decomposed into sub-activities, which can be analyzed in terms of gestural movements, poses, and postural transitions, as well as using a series of cognitive dimensions such as activity planning. Since this evaluation is qualitative, it can vary based on the person being monitored and

the caregiver's experience. A significant amount of research work is implicitly or explicitly aimed at augmenting the health-care domain-expert's qualitative evaluation with quantitative data or knowledge obtained from HAR.

From a medical perspective, there is still a lack of evidence about the technology readiness level of smart home architectures supporting older persons by recognizing ADL [2]. We hypothesize that this may be due to a lack of effective collaboration between smart-home researchers and health-care domain experts, especially when considering HAR. We foresee an increase in HAR systems being developed in close collaboration with caregivers and geriatricians to support their qualitative evaluation of ADL with explainable quantitative outcomes of the HAR systems. This has been a motivation for the work in this thesis. The recognition of human activities – in particular ADL – may not only be limited to support the health and well-being of older people. It can be relevant to home users in general. For instance, HAR could support digital assistants or companion robots to provide contextually relevant and proactive support to the home users, whether young adults or old. This has also been a motivation for the work in this thesis.

1.1.2 Challenges and problem description

Depending on the application or motivation, researchers would like to have a certain set of features in the HAR systems they develop. In the literature, one can see researchers attempting to incorporate features such as (i) scalability of the HAR system in terms of accommodating more number of heterogeneous sensors and more variety of activities that can be recognized, (ii) online activity recognition which means computational performance must be taken into consideration, (iii) accuracy of activity recognition, part of which is also the ability to deal with uncertainty-of and noise-in sensory data, (iv) intelligibility in terms of explainable results and inner-workings of the system, (v) learning ability in order to improve HAR accuracy over time, and (vi) considerate to privacy. It is noteworthy that within the literature each feature is a research problem [6].

Given the motivations described in Section 1.1.1, namely, (i) facilitation of iterative development and ease in collaboration between HAR system researchers/developers and health-care domain experts in ADL, and (ii) robust HAR that can support digital assistants or companion robots. There is a need for the development of a HAR framework that at its core is modular and flexible to facilitate an iterative development process [3], which is an integral part of collaborative work that involves develop-test-improve phases. At the same time, the framework should be intelligible for the sake of enriched collaboration with health-care

domain experts. Furthermore, it should be scalable, online, and accurate for having robust HAR, which can enable many smart-home applications. The goal of this thesis is to design and evaluate such a framework.

1.2 Contributions

In this section, all research contributions of this thesis are introduced. Note that these contributions have been achieved in collaboration with my research group, which is TheEngineRoom [1] at the University of Genova (Italy). Nonetheless, in this section, I also list my specific significant contributions although the proposed contributions are the result of teamwork. In this thesis, three contributions are made to the field of HAR. The first contribution is Arianna+, a knowledge-based framework for contextualized HAR. The second contribution is OWLOOP, an Application Programming Interface (API) that supports the implementation of HAR systems based on Arianna+. The third contribution is the evaluation of a set of HAR systems based on Arianna+ and developed using OWLOOP API. The results of the evaluations emphasize the novelty of Arianna+.

1.2.1 First contribution

Chapter 3 presents Arianna+, a framework to develop networks of ontologies - for knowledge representation and reasoning - that enables smart homes to perform human activity recognition online. In the network, nodes are ontologies that represent different contexts and allow data contextualization, while edges are computational procedures that process data. The framework allows orchestration of data within a network of ontologies such that the orchestration results in contextualized HAR.

Being a knowledge-based framework, Arianna+ is intrinsically intelligible and we further argue that a network of small ontologies is even more intelligible. The framework allows to integrate within the same architecture heterogeneous data processing techniques as computational procedures. Thus, with Arianna+, we do not propose a new algorithmic approach to HAR, instead, we present a modular knowledge-based architectural approach to accommodate knowledge-based and data-driven activity models in a context-oriented way. This is a novelty of Arianna+ as compared to other works in the literature. Chapter 3, presents in a detailed manner Arianna+ and its core components. Later, chapter 5 evaluates Arianna+. It showcases the scalability of systems based on Arianna+, their ability of online

[1] https://theengineroom.dibris.unige.it/

activity recognition, intelligibility, and potential for good HAR accuracy. This chapter is based on the following publication, which is currently – at the time of writing this thesis – already published. The Table 1.1 clarifies the contribution of each author with respect to the publication.

*"L. Buoncompagni, **S. Y. Kareem** and F. Mastrogiovanni,"Human Activity Recognition models in Ontology Networks", in IEEE Transactions on Cybernetics, (2021). DOI: https://doi.org/10.1109/TCYB.2021.3073539"*

Authors	Contributions	Keywords
Luca Buoncompagni	Writing (original draft) Conceptualization Experiments Investigation	Introduction Formalism Experimental design Implementation Evaluation
Syed Yusha Kareem	Writing (original draft) Conceptualization Investigation	Formalism Related work Evaluation Discussion
Fulvio Mastrogiovanni	Writing (review/editing) Conceptualization Supervision	Discussion Reviewing

Table 1.1 Human Activity Recognition Models in Ontology Networks.

1.2.2 Second contribution

Chapter 4 presents OWLOOP, an API that supports the development of HAR system architectures based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object-Oriented Programming (OOP). OWLOOP allows to construct and use Descriptors, which are Java classes that interface OOP objects with knowledge structured in program-memory as an ontology. Descriptors encapsulate boilerplate code to simplify the development and maintenance of a system that exploits knowledge representation and reasoning using ontologies. The Descriptors' methods allow to read, write, update, delete, and reason on axioms in ontology files. Furthermore, flexibility in Descriptor construction allows to avoid drawbacks in computational performance. This chapter is based on the

following publication, which is currently – at the time of writing this thesis – under review. The Table 1.2 clarifies the contribution of each author with respect to the publication.

*"L. Buoncompagni**, S. Y. Kareem** and F. Mastrogiovanni, "OWLOOP: A modular API to describe OWL axioms in OOP objects hierarchies", in SoftwareX, (2021).**Equal contribution*."

Authors	Contributions	Keywords
Luca Buoncompagni	Writing (original draft) Conceptualization Experiments	Implementation Maintenance
Syed Yusha Kareem	Writing (original draft) Investigation Experiments	Evaluation Refactoring Maintenance
Fulvio Mastrogiovanni	Writing (review/editing) Supervision	Reviewing

Table 1.2 OWLOOP: A modular API to describe OWL axioms in OOP objects hierarchies.

1.2.3 Third contribution

Chapter 5 presents the evaluation and exploitation of Arianna+ using OWLOOP API. It has four sections. Each section presents a HAR system based on Arianna+ and developed using OWLOOP API. Section 5.1 evaluates the computational performance of a HAR system based on Arianna+. Preliminary results show that HAR done in a contextualized manner using a network of ontologies is computationally better compared to HAR done in a non-contextualized manner. Results, therefore, highlight a HAR system that is intelligible, online, and scalable.

Section 5.1 in Chapter 5, is based on the following publication, which is currently – at the time of writing this thesis – already published. The Table 1.3 clarifies the contribution of each author with respect to the publication.

*"**S. Y. Kareem**, L. Buoncompagni and F. Mastrogiovanni,"Arianna +: Scalable Human Activity Recognition by reasoning with a network of Ontologies", in proceeding of the 17th International Conference of the Italian Association for Artificial Intelligence (AIxIA), Springer Cham, pages 83-95, (2018). DOI: https://doi.org/10.1007/978-3-030-03840-3_7"*

Authors	Contributions	Keywords
Syed Yusha Kareem	Writing (original draft) Conceptualization Experiments Investigation	Introduction Related work Experimental design Implementation Evaluation
Luca Buoncompagni	Writing (review/editing) Supervision	Discussion Reviewing
Fulvio Mastrogiovanni	Writing (review/editing) Conceptualization Supervision	Discussion Reviewing

Table 1.3 Arianna +: Scalable Human Activity Recognition by reasoning with a network of Ontologies.

Section 5.2 in Chapter 5 evaluates a HAR system (based on Arianna+) that has a network of ontologies wherein the spatial context is shared among all activities and temporal context is specialized for each activity. Activity models are developed to recognize activities being performed in the CASAS dataset. Results show that contextualized activity recognition has accuracy comparable with non-contextualized state-of-the-art data-driven and hybrid approaches. Results, therefore, highlight a HAR system that has good accuracy (for the particular CASAS dataset), is intelligible, and is online. This section is based on the same publication mentioned in Section 1.2.1 as the first contribution. This publication is currently – at the time of writing this thesis – under review after a second revision. This section presents all the practical aspects of that publication, i.e., experimental design, evaluation, results and discussion. The Table 1.1 clarifies the contribution of each author with respect to the publication.

*"L. Buoncompagni, **S. Y. Kareem** and F. Mastrogiovanni,"Human Activity Recognition models in Ontology Networks", in IEEE Transactions on Cybernetics, (2021). DOI: https://doi.org/10.1109/TCYB.2021.3073539"*

Section 5.3 in Chapter 5 exploits Arianna+ to develop a real-world HAR system that can recognize the routine morning hygiene activity and having breakfast activity. Both are composite activities. In the network of ontologies for this HAR system, there is an ontology in which the results of HAR are saved. These results can be queried in an intelligible manner using SPARQL protocol and Query Language (SPARQL). Results highlight the intelligibility of the HAR system. This Section is based on work that has not been published yet. My

contribution in this work has been the following: conceptualization, literature research, experimental design, implementation, and evaluation.

Section 5.4 exploits Arianna+ to develop a real-world HAR system that can recognize having breakfast activity. The HAR system is integrated with a dialogue management system and a digital assistant. Once the user's having breakfast activity is detected, the digital assistant reminds the user to take medication. The user experiences this scenario with two kinds of assistants, i.e., one with a robot body and one without a robot body. A comparison is made between the UX when a user interacts with the two kinds of digital assistants. Results show that the digital assistant with a robot body is comparatively perceived as more attractive, stimulating, and novel. With this preliminary result, we argue that by providing a robot as a companion digital assistant the acceptance of a smart home system could be positively affected. This Section is based on the following article, which is soon to be submitted to the International Journal of Social Robotics. Table 1.4 clarifies the contribution of each author with respect to the article.

*"T. Elia, **S. Y. Kareem** and F. Mastrogiovanni,"A smart home system with a digital assistant - Comparison between a vocal-assistant and a robot-assistant", in the International Journal of Social Robotics, (2021)"*

Authors	Contributions	Keywords
Tommaso Elia	Writing (original draft) Experiments Investigation	Experimental design Implementation Dialogue management Digital assistants (vocal/robot) Survey and Evaluation
Syed Yusha Kareem	Writing (original draft) Experimental setup Supervision	Implementation Arianna+ and OWLOOP API Dialogue management Evaluation and Reviewing
Fulvio Mastrogiovanni	Writing (review/editing) Conceptualization Supervision	Experimental design Reviewing

Table 1.4 A smart home system with a digital assistant - Comparison between a vocal-assistant and a robot-assistant.

1.3 A brief guide to reading this thesis

This section outlines the structure of this thesis and presents the flow between all the chapters.

Chapter 1: Introduction

Introduces the reader to the domain of HAR in smart environments. It presents particular motivations for HAR in smart-homes. It then briefly describes the challenges of doing HAR. Then, based on the motivations and challenges, it presents the goal and contributions of this thesis.

Chapter 2: Related work

Presents the background that is necessary for understanding the approach taken in this thesis. The background includes an introduction to the key terminologies in the domain of HAR for health and well-being in smart-homes. Furthermore, the chapter presents the state of the art on sensing and modeling approaches for HAR. Also, the benefits and drawbacks of these approaches are clarified. Then, based on the challenges within the state of the art and the motivations mentioned in Chapter 1, the research problems that are particularly addressed by this thesis are highlighted.

Chapter 3: Arianna+

Presents Arianna+, a framework that has been designed taking into account the research problems highlighted in Chapter 2. The first contribution of this thesis is Arianna+, a modular and intelligible framework for robust HAR. This chapter presents the preliminaries that are necessary for understanding the core components of Arianna+. The chapter then highlights the novelty of Arianna+ and describes it before presenting in detail all its core components.

Chapter 4: OWLOOP API

Presents the second contribution of this thesis that is the OWLOOP API. It is an API that is designed to ease the development of HAR systems, which are based on Arianna+. It maps OWL axioms into OOP objects called Descriptors. This chapter presents the preliminaries that are necessary for understanding the core components of OWLOOP API. The chapter then highlights the novelty of OWLOOP API and describes its OWL to OOP mapping approach before presenting in detail all its core components and its usage in HAR system development.

Chapter 5: Evaluating and exploiting Arianna+ using OWLOOP API

Presents four implementations of HAR systems that are based on Arianna+ and are developed using OWLOOP API. Each implementation is evaluated and its results are presented highlighting how they address one or more of the research problems described in Chapter 2. Thus affirming the novelty of Arianna+. In this chapter, each implementation is presented as

a section with its own introduction, description of the methodology, experimental evaluation, and summary.

Chapter 6: Conclusion

Finally, presents a summarized cohesive conclusion by recapping the three contributions of this thesis with respect to the initial research problems presented in Chapter 2. The chapter then presents limitations of the HAR approach proposed in this thesis and opportunities of future work, i.e., the open issues and promising research directions.

Chapter 2

Related work

2.1 HAR for health and well-being in smart homes

Scientific and technological progression has progressively improved the standard of living for the people. Research findings have shown that in recent years, due to a decrease in birth rates and an increase in life expectancy, a demographic shift will occur. The population of the elderly is predicted to grow between the years 2020 to 2050. In 2020, 9.3% of the population is aged 65 or older, and by 2050, that figure is predicted to be closer to 16.0% [1]. To protect and improve the quality of life of older people, and to respond to the needs of the increasingly aging population, the problem of facilitating stable and active aging must be addressed. Advances in modern technology have provided new opportunities to enhance the standard of independent living for older people. Ambient Assisted Living (AAL) services are being built to help residents track and protect their safety and well-being. The aim of AAL systems is therefore to provide sufficient support to allow older people to live in their homes as independently as possible for as long as possible. The AAL system needs to understand a person's behaviour to be able to provide such assistance, so it relies on HAR systems to identify and interpret sensory data. A smart home is characterized in [14] as an environment equipped with sensors to track the activities of residents and their interactions with the environment by analyzing the data from the sensors.

HAR Literature indicates that the term activity was used with minor variations. There is, however, a notion that emerges from the widespread usage of the word. The terms action and activity are also used interchangeably in the literature. However, a distinction is made between action and activity based on granularity [15]. On the basis of such granularity, complex activities (e.g. "preparing tea") are hierarchically organized into simpler activities or actions (e.g., " enter the kitchen", "near water kettle", "pouring "). Activities can be classified

into composite activities, sequential activities, concurrent activities and interleaved activities, taking into account the complexity, duration and order of the sub-units. For instance, in the case of sequential activity, the following sequence of sub-units can make up the sleeping activity: open the door, lie on the bed, and turn off the light. In the case of interleaved activities, a person prepares a meal, then takes a break and goes to the bathroom, then returns to the preparation of the meal. Lastly, in the case of concurrent activities, activities such as watching TV and eating may be carried out concurrently by the user.

Context data, such as location, time, frequency, and object interaction, are also essential to the identification of activities in smart home settings. Location refers to a physical region where a given activity is often carried out. Taking a shower, for example, takes place in the bathroom. Human posture is almost always related to the location of the person and often to furniture. Some primary postures, such as sitting and lying, are used to recognize other activities, such as resting or sleeping. Time, which involves the start and end times (or duration), is another key feature for defining activities. For example, long sleep typically happens in the evening and in a semi-regular period of time within a typical daily routine. Finally, when monitoring human activities, contact with objects is generally considered, where the object may signify the relevant activity being conducted. For example, the use of a broom as an object can imply ongoing housekeeping activity. Other contextual dimensions, such as temperature and humidity, may also be considered for activity detection. These features are useful for identifying the activities and determining the health status of older people in accordance with health domain experts.

Many interpretations of the term context have been proposed in other studies within the literature. Context is, as it is described by [16], any information that can be used to characterize an entity's situation. An entity shall include a person, a place, or an object. The authors of [17]: "raw data and background information", make a distinction between two forms of information. On the one hand, raw data, also called low-level context data, are unprocessed data taken directly from the source, such as a sensor. Such data can reflect a person's vital signs or movements or environmental parameters (e.g., temperature, humidity, and sound). Context information, on the other hand, is information produced by the processing of raw data. It refers to the extraction of high-level data such as the activity patterns of a person. Context plays a vital role in the monitoring of health and wellbeing. In the case of smart environments delivering health care services, the assessment of the health status of monitored individuals and their success in achieving their day-to-day activities relies on knowing their surroundings. For example, changes in the data on vital signs of the subject should be correlated with the current situation of the subject to gain a better

understanding of the condition of the subject. In a smart environment, data regarding the person's context (physiological, temporal and spatial) is often collected using a variety of sensors and devices. To ensure the efficiency of the smart environment, it is necessary to ensure that the appropriate sensors are properly selected and that the data is appropriately and accurately analyzed [18].

2.1.1 Activities of Daily Living (ADL)

Recognition of Activities of Daily Living (ADL) is the recognition of day-to-day activities in an indoor environment such as a home. These activities include eating, cooking, sleeping, sitting, bathing, dressing, grooming, etc. Recognition of such activities is of high significance for their implementation in a number of fields, such as smart homes. Recognizing the daily activities of older people living independently at home will help caregivers track their wellbeing and provide adequate treatments. In addition, a smart home may provide contextual and constructive assistance to home users if it identifies their activities. ADL is categorized as two [2], basic ADL (BADL) and instrumental ADL (IADL). BADL includes tasks such as grooming, eating, climbing up and down the stairs, dressing and mobility (walking), while IADL includes activities such as ironing, sweeping, washing dishes and recreational activities (e.g., watching TV). IADL is somewhat different in that it needs a certain level of planning capability and sometimes social skills, such as housekeeping, cleaning and cooking.

A variety of solutions have been suggested to identify ADL over the last decade. Some strategies use surveillance cameras to capture images or video, and then use computer vision techniques to identify the activities conducted [19, 20]. Other techniques are based on wearable sensors and object-tagged sensors such as accelerometers, motion sensors, pressure sensors, and Radio Frequency Identification (RFID) tags for the detection of everyday activities. The authors of [21] suggested a technique focused on wrist-worn sensors to identify the behaviors of older people to support independent living. Three types of sensors are attached to the user's wrist-worn watch: accelerometer, temperature sensor and altimeter. A similar technique was developed in [22] for the identification of housekeeping activities, using accelerometer and gyroscope as wrist-worn sensors. In [23], a technique was proposed that uses an accelerometer as a wrist-worn sensor, and items of everyday use are marked with RFID tags. The authors tested their methodology in three ways: using data only from the accelerometer, using data only from RFID tags, and using data from both the accelerometer and RFID tags. The findings show that the hybrid approach (i.e. integrating data from both accelerometer and RFID tags) produces better results compared to separate approaches. A

similar solution has been presented by [24], in which the user has to wear a system consisting of an accelerometer, a gyroscope, a magnetometer and an RFID antenna. Various items of everyday use are also tagged with RFID tags. The authors tested the proposed framework in two scenarios: breakfast (preparing and having breakfast, washing dishes, etc.) and home care.

In addition, several other techniques have used dense sensing and utilized various sensors, such as motion sensors, pressure sensors, temperature sensors and proximity sensors, have been deployed in the environment [25]. When a user conducts any activity in the vicinity of these sensors, relative information can be collected by means of these sensors, which can then be used for activity recognition purposes. In [26], the authors proposed a technique called SmartWall that uses passive RFID tags attached to a wall. When users perform some activity in front of this wall, the changes induced by the reflected signal capture information about the activity performed. The machine learning algorithm used is based on a multivariant Gaussian algorithm that uses the maximum likelihood estimation to identify activities. The proposed system can identify 10 daily activities and can also detect falls. The authors carried out a prototype of the proposed solution and carried out various experiments to evaluate the performance. Apart from recognizing activities, in [27], the authors present a methodology to represent the habits of users based on the sequence and duration of their activities. The uniqueness of their approach is that they identify human habits using the assessment of sensor signals by clustering.

2.1.2 Ambient Assisted Living (AAL)

It is up to society to provide for the health and well-being of older people as the population grows older. Many elderly people prefer to live peacefully in their own homes. A considerable amount of research has been undertaken in recent years to provide home health monitoring and healthcare services. Researchers have developed a range of emerging technology to support people with day-to-day activities, dubbed ambient intelligence. One of these technologies is called Ambient Assisted Living (AAL). It aims to provide services such as remote health tracking, medication management, prescription reminder, exercise management, and constructive support for independent living. Over the last decade, a variety of solutions have been proposed under the umbrella of AAL to promote the independent living of older people [28–30]. HAR plays a preliminary and important role in most of the solutions. As described in [31, 32], depending on the type of sensors used in the smart

home, HAR may be performed using data from, vision, wearable or distributed sensors, or a combination of these.

A vision-based approach uses a surveillance camera to gather information on the activities of residents [33]. Several other solutions are also suggested using different sensors. The authors of [34] suggested a multi-sensor strategy for the identification of day-to-day activities in robot-assisted living. In [35], the authors proposed an RFID-based framework to track the activities of Alzheimer's patients at home. The basic concept of this work is to monitor a user's movement from one room to another and to report any irregular circumstance (e.g., staying in the washroom for a longer time). Some solutions have tried to incorporate robotics into smart home for AAL. Domestic assistance has been a driving force in the area of mobile robotics so that robots can support people in their everyday environment. Mobile robots can be very useful to help elderly people live independent lives. Over the decades, several mobile robots have been built by academics and research groups. Their findings and observations, obtained through their studies, would certainly shape tomorrow's care robots. Among all mobile robots for the health care and well-being of older people, there are many notable robots mentioned in [36].

2.2 Sensing approaches for HAR

2.2.1 Visual sensor-based

This section presents some works that concentrate on based visual sensor based solutions for HAR. In [37], the authors presented a survey of current research work that uses a visual sensor-based approach for activity detection and divided the literature into two major categories: uni-modal and multi-modal approaches. Unimodal methods use data from a single modality and are further classified into stochastic, rule-based, space-time-based, and shape-based methods. Multimodal techniques use data from various sources and are further categorized into behavioral, effective and social networking methods. In addition, the authors of [38] presented a detailed description of the research conducted in the field of action recognition using vision-based approaches. This survey categorizes the overall work into two main categories: representation-based solutions and deep neural network solutions. Representation-based solutions are further categorized in Holistic and Local Representations. Deep neural network solutions are subclassified as multiple stream networks, temporal coherency networks, generative models, and spatiotemporal networks. One of the problems with conventional cameras is the reliance on light, i.e. they can't function in the dark.

Developing a depth camera like Kinect has solved this problem because it can operate in the dark. It is possible to obtain different data streams from Kinect, such as RGB, depth, and audio [39]. Kinect data can collect details about the human body and create a virtual 3D skeleton. Using this information, activities can be identified since different body (skeletons) movements are connected to different activities. Apart from complex computing, the cost of depth sensing cameras is high, which is a drawback of this method. While some sensors (e.g. cameras) have high precision for activity or action recognition, simpler sensors (e.g. Passive Infrared (PIR), light, and RFID) are commonly used in smart home environments due to privacy concerns.

2.2.2 Non-visual sensor-based

Non-visual sensors are often wearable sensors, or object-tagged sensors, or ambient/distributed sensors. Wearable sensors and object-tagged sensors are typically accelerometers, gyroscopes, magnetometers, and RFID readers and tags. Whereas ambient sensors are commonly motion sensors, pressure sensors, and proximity sensors. Wearable sensors typically refer to sensors that are directly or indirectly mounted on the human body. They produce signals when the user conducts an activity. As a result, they will track features that are descriptive of the physiological state or movement of the individual. Wearable sensors may be mounted in clothing, eyeglasses, belts, shoes, wristwatches, mobile devices or placed directly on the body. They can be used to gather information such as body position and movement, pulse, and skin temperature.

Researchers have found that different types of sensor information are effective in classifying different kinds of activities. Accelerometer sensors are perhaps the most widely used wearable sensors for activity tracking. They are especially successful in tracking acts involving repetitive body movements such as walking, running, sitting, standing, ascending stairs. In [40], the authors include a description of the research work that recognizes human behaviors using acceleration data. In [41], a network of 3-axis accelerometers is distributed around the body of the user. Each accelerometer provides detailed information on the orientation and movement of the corresponding body part. [42] identifies human ambulation and posture using acceleration data obtained from the hip. Researchers have also collected multi-sensory datasets for everyday life activities [43, 44]. Wearable sensor-based activity tracking suffers from limitations. Most wearable sensors need to run continuously and work hands-free. This may have difficulties in the real-world situations of application. Practical concerns include the acceptability or willingness to use wearable sensors and their

viability and ability to wear them. Real-world technological issues include size, ease of use, battery life, and the approach's effectiveness. In order to fix these problems, extensive research on smart garments has been carried out, which is focused on embedding sensors in clothing for HAR [45]. Another ongoing research thread seeks to use common consumer products that people carry everyday, such as smartphones, as intelligent sensors for activity tracking, identification, and assistance. This has been in effect for some time and is expected to increase dramatically as a result of the latest growth and cost reduction of palm-held electronic devices.

Wearable sensors are inappropriate for complex bodily movements and/or multiple interactions with the environment. Data from sensors worn by the consumer may not be adequate in some cases to distinguish activities that are simply physical movements (e.g., brushing teeth and washing dishes). As a consequence of this, dense sensing or ambient sensing has arisen. Dense sensing-based activity tracking requires the use of multiple sensors to connect to various objects in an area, and then activities are tracked by detecting when the sensors sense user-object interactions. This strategy is focused on evidence from the real world, which demonstrate that activities are defined by the artifacts used during their execution. A clear sign that an item is being used may also be an invaluable clue to the activity that is going on. As such, it is believed that behaviors can be identified from sensor data that tracks human interactions with objects in the environment. By dense sensing, we are referring to the way and scale in which the sensors are used. Using dense sensing, a large number of sensors, typically low-cost low-power and miniaturized, are deployed in a variety of objects or positions within an area to track the movement and activity of a person. As dense sensing-based monitoring embeds sensors in environments, this makes it more suitable for building smart environmental applications such as smart environments. As such, dense sensing-based activity tracking has been widely implemented in AAL via the smart home framework. Sensors in a smart home can track a person's movements and environmental events in such a way that the smart home HAR system can infer ongoing activities based on sensor observations, thereby providing just-in-time context-aware ADL assistance. For example, a switch sensor in the bed may strongly indicate sleeping.

It is worth noting that wearable sensors and dense sensing are not mutually exclusive. They have to work together in some applications. For example, RFID (Radio Frequency Identification) based activity monitoring requires items to be tagged and users to wear an RFID reader attached to a glove or bracelet. In [46, 47], the authors created two devices, iGlove and iBracelet, acting as wearable RFID readers to detect when users communicate with unobstrusively tagged objects. The authors of [48] performed fine-grained activity

recognition (i.e. not only recognizing that a person is cooking, but deciding what they are cooking) by aggregating the use of items. In [49], authors identify indoor day-to-day activities by using an RFID sensor network. In most cases, wearable sensors and dense sensors are complementary and can be used in combination to achieve optimum recognition performance. For example, [50] combines wearable sensors and object sensors for the collection of multimodal sensor information. They identify sequential, interleaved, and concurrent activities via a pattern-based approach. Although a large amount of research has been done and advances have been made, the research is still very active when using wearable sensors and dense sensing techniques.

2.3 Modeling approaches for HAR

There are two major types of ADL recognition methods: data-driven methods and knowledge-based methods [51]. Data-driven approaches are more versatile regarding implementation, and they are more resilient to noisy and ambiguous sensory data. Additionally, since they do not depend on a static specification of how ADL should be done, they may theoretically account for more variations of the different activities considered. A good example is when preparing a meal, a person may begin by turning on the stove and then retrieve the food items from the cabinets or vice versa. The person can choose to skip or add some steps to the activity, depending on the recipe. Therefore Data-driven approaches are more flexible in terms of how activities or actions are executed. However, obtaining a robust annotated dataset is costly and often infeasible. It is also difficult to provide the domain knowledge about the activity using data-driven methods. Knowledge-driven approaches, on the other hand, are more capable of reflecting the semantics of sensor events. These techniques exploit domain knowledge to model a real-world activity. An activity can be modeled this way, removing the need for a broad training dataset. Such strategies, however, lack the advantage of system versatility in terms of the range of variability in the activities performed. Besides the methods described above, another alternative that has been suggested is to combine data-driven and knowledge-based approaches to form hybrid approaches. In the following, we go through these categories in greater depth.

2.3.1 Data-driven

Data-driven approaches use sensor data, labeled with previously performed activities, as a training set, and machine learning algorithms to create a model for inferencing activities.

These models can be created u sing sensory data streams [52] or event-based data [53]. A discriminative (i.e., Support Vector Machines [53] and Artificial Neural Networks [54]) data-driven approach is utilized in situations with complex, multi-modal data streams, for example, data originating from cameras [55] and accelerometers [56], for posture recognition and fall detection. A generative (e.g., Hidden Markov Models [57] and Dynamic Bayesian Networks [58]) data-driven approach is used in circumstances where simple data is at play (e.g., while using distributed sensors such as pressure sensors, PIRs, etc.)." Observations concerning the surrounding environment of an individual (in particular the use of objects), possibly coupled with body-worn sensor data, become the basis for HAR systems [23, 59]. In [60], the authors have proposed a time series data analysis approach to classify ADL using sensor events as a series. The application of Hidden Markov Model inference is proposed in [61] to identify activities based on features extracted from recent sensor events using a sliding window. Another method for detecting sequential, interleaved, and concurrent activities is Conditional Random Fields proposed in [62]. To specifically model complex temporal dependencies over time intervals, the authors in [63] combine Bayes Networks with Allen's interval algebra [64]. In [65], the authors introduce a supervised learning classifier that dynamically updates its model in response to dynamically discovered context. However, as training data is difficult to obtain in practical settings, systems that rely on supervised learning are vulnerable to scalability problems as more events and more background data are considered. Datasets of complex ADL are coupled closely to the context in which they are obtained (i.e., the home environment and sensor setup), as well as to the manner in which the activities are conducted by a particular individual. Because of this, portability of activity datasets is still an open issue [66].

2.3.2 Knowledge-driven

Knowledge-based approaches are based on specification-based definitions of the characteristics and semantics of complex activities. Matched with available sensor data, these are used to classify the current activity. Such definitions are described by logical axioms, rules, or description logics [67–69]. Much research has also gone into accommodating time related domain knowledge into the reasoning process [70–73]. Some research has also employed concepts drawn from constraint-based planning in conjunction with temporal reasoning techniques [74, 75] for activity recognition. In the following paragraphs, we particularly deeply explore various knowledge representation and reasoning frameworks that have been built to model complex human activities using description logic (DL) based ontologies.

In a knowledge-based approach to reasoning on contextualized data (i.e. sensor events) correlated with a priori defined knowledge (i.e., schema, collection of axioms, laws, etc.), the Ontology Web Language (OWL) standard has been widely proposed for HAR, as discussed in [76]. In particular, description logic [77] has appeared, among other symbolic formalisms, mainly because it offers a complete justification (i.e. any true well-formed formula can be derived) backed by reasoning tools. Description Logic (DL) [77] is a state-of-the-art formalism that allows representation and reasoning of symbolically organized knowledge. DL is part of a decidable fragment of languages that are based on first-order logic and has been standardized as OWL-DL by the World Wide Web Consortium (W3C). In order to accommodate temporal representations of knowledge, both probabilistic and logic-based methods utilize Allen's Interval Algebra [64]. OWL [78] enables the creation of ontologies, which are semantic corpora containing knowledge encoded as logically specified symbols intended to be readable by humans and machines. An ontology is described in a context, i.e. a coherent corpus of knowledge representing a domain of interest [16]. Within a context, a reasoner can assess the consistency of knowledge by performing subsumption, instance-checking and inference on the basis of well-defined rules [79]. Several reasoners have been proposed, some of which are discussed in [76]. Deterministic DL reasoners (e.g., Pellet [80]) do not reflect ambiguity, whereas fuzzy [81] or probabilistic [82] OWL reasoners do.

Various research works relied on description logic languages to formally express activity definitions [68, 69]. Context information of ADL was used to construct activity models used to identify ADL based on the similarity of the sequences of sensor events to the general models [60]. Ontological reasoning has also been proposed for performing dynamic segmentation of sensor data [83–85] or for refining the output of supervised learning methods [86]. Defeasible reasoning has been adopted to improve current sequential activity recognition systems by detecting interleave activity and managing consistent or conflicting information [87]. A further framework to segment activities based on their semantic definition is proposed in [83]; this method also supports the recognition of concurrent activities. However, these works are based on static assumptions regarding simpler components (i.e. sub-units) of activities. Thus, while the specification-based approach is successful for activities defined by a few standard execution patterns, it is hardly scalable for the extensive specification of complex ADL in various real-world contexts. In addition, DL is computationally costly, and this restricts its online use, particularly when the context is information rich. In particular, computing scales exponentially with the amount of information encoded in ontology [77].

While we are discussing the subject of knowledge-based approaches it is worth understanding the differences and similarities in the words 'ontology', 'knowledge-base', and

'knowledge-graph'. In [88], the authors attempt to clarify these words based on an extensive literature review. They highlight that ontological representations allow semantic modeling of knowledge, and are therefore commonly used as knowledge bases in artificial intelligence (AI) applications, for example, in the context of knowledge-based systems. Furthermore, the authors explicitly emphasize that an *ontology* does not differ from a *knowledge base*, although ontologies are sometimes erroneously classified as being at the same level as database schemas [89]. This is because in fact, an ontology consists not only of classes and properties (e.g., owl:ObjectProperty and owl:DatatypeProperty), but can also hold instance data (i.e., the population of the ontology). Regarding the terms *knowledge-graph* and *ontology*, the authors point out that the difference between a knowledge graph and an ontology could be interpreted either as a matter of quantity (e.g., a large ontology), or of extended requirements (e.g., a built-in reasoner that allows new knowledge to be derived). They highlight that the second interpretation leads to the assumption that a knowledge graph is a knowledge-based system that contains a knowledge base (e.g., an ontology) and a reasoning engine. Based on these aspects, the authors define knowledge-graph as follows: '*A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge.*'.

2.3.3 Hybrid

In view of the limitations of both data-driven and knowledge-based methods, a few hybrid behavior recognition systems have been proposed in the literature that differ on the adopted reasoning techniques and their interaction mechanisms. An interesting example of these methods is Markov Logic Networks (MLN), a probabilistic first-order logic method [90]. Given a training set and the set of probabilistic formula, with MLN it is possible to learn weight for each grounded formula by iteratively optimizing a pseudo-likelihood measure. These weights are the confidence value of the formula. Deterministic formulae can be applied to probabilistic formulae to express deterministic knowledge of the domain of interest. Various reasoning tasks can be carried out in order to infer additional knowledge on the basis of formulas and facts [91]. A similar approach was adopted in [92] to model and identify activities at various levels of complexity using probabilistic description logic.

The benefit of using probabilistic logic is that it allows the definition of complex knowledge-based constraints that can capture the inherent uncertainty of sensor measurements. Learning the weights of these constraints makes it possible to combine a strong point of knowledge-based and data-driven approaches, thereby improving the recognition rate. However, these hybrid methods also involve the acquisition of a labelled data set. Hybrid

logic-statistical frameworks have been suggested in the literature. For example, activities are defined with a combination of probabilistic models and semantic constraints in [92–94]. Other contextualization strategies use a hybrid solution through augmentation of ontologies with specific ML-based algorithms, which are based on patterns of activity [95, 96], or by inferring concepts through a learning approach [97]. In [98], an ontology is used to classify features that are useful for the training of human activity models, while probabilistic ontology is used for the same task in [92]. MLN is extended by Allen's Algebra in [99] and is adopted in [100] for the purpose of refining structures designed through knowledge engineering. In [101] the authors suggested that ontologies be used to extract semantic similarity between sensor events. This similarity is then used for segmenting sensor data, obtaining sequential activities' patterns that are then used to train a clustering model. Semantic segmentation of sensor data enables accurate individuation of transitions between activities without supervised techniques. The main downside of this approach is that it involves a comprehensive dataset of activities (even if not labelled) obtained from the tracked subject to create an accurate activities' model. Hybrid ontological and statistical rationale is also proposed in [102] to continuously assess the risk of falling of an older person at home by combining data acquired from various fall detection systems and environmental sensors. Semantic reasoning, given the context, is then used to minimize the number of false positives obtained by the statistical fall-detection system.

Context-aware methods have shown to be successful in assessing human activities if they are carried out on data semantically organized in an ontology. Effectiveness depends on the prior information that is used to organize data, and this contributes to a significant effort in knowledge engineering. Indeed, seeking a conceptualization that perfectly represents the context to support the recognition of human activity is far from trivial. Several ontology-based representations have been evaluated in [97] based on the context they describe, e.g. sensor events, sensor hierarchies, human postures, or locations. In [94], the context is described in terms of affordances, and is based on relevant knowledge of space, time and human attitudes. In these methods, the need for efficient reasoning on data generally leads to ontologies that are built either for computational efficiency or for high expressivity, i.e. they accurately describe human activity in detail. For example, the work in [103] reports an ontology with restricted expressivity, based on OWL-EL, which has been implemented in a real-world use case. That work exhibits the trade-off between the computational effort required for reasoning and the expressivity of knowledge representation.

2.4 Research problems addressed by this thesis

In this section, we outline the research questions (RQs) tackled in this thesis. For each question, we introduce the research problem and indicate the specific chapter of the thesis where the problem is addressed.

RQ1. How can one have a HAR system that at its core is modular and intelligible – to support an iterative development process of such a system and for an enriched collaboration between HAR researchers and health-care domain experts?

As highlighted in Section 1.1.1, there is still a lack of evidence about the technology readiness level of smart home architectures supporting older persons by recognizing ADL [2]. We hypothesize that this may be due to a lack of effective collaboration between smart-home researchers and health-care domain experts, especially when considering HAR. As can be seen in Section 2.3, whether data-driven, knowledge-based, or hybrid, most researchers have taken an algorithmic approach towards activity recognition. Some researchers have taken an architectural approach towards health monitoring and a survey categorizes them into centralized architectures and distributed architectures [31]. Nonetheless, modularity remains an open issue in the domain of HAR.

Hence, to support an iterative development process and an effective collaboration between HAR researchers and health-care domain experts, Chapter 3 presents Arianna+, a framework for developing modular and intelligible HAR systems. Intelligible because the framework is designed based on the knowledge-based approach. Particularly utilizing Description Logic (DL) of Ontology Web Language (OWL) and Semantic Web Rule Language (SWRL) rules. Modular because the framework takes an architectural approach that allows contextualized knowledge representation and allows to incorporate data-driven and knowledge-based reasoning techniques, and consequently it can incorporate heterogeneous sensors. Chapter 3 highlights these aspects while presenting the core components of Arianna+.

Often, the development and programming aspects, which are associated with the research in HAR systems, are neglected. They play an important role in an iterative development process [3], which is an integral part of collaborative work. Hence, chapter 4 presents OWLOOP, an API that supports the development of HAR system architectures based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object-Oriented Programming (OOP). Chapter 4 presents OWLOOP's novelty and core-components.

RQ2. If a knowledge-based approach is taken for ADL recognition then how can it be done in a scalable and online manner?

It is known in the literature that DL-based reasoning is computationally expensive, and this limits its online usage, especially when the context is information-rich. In particular, the computation scales exponentially with the amount of knowledge encoded in an ontology [77]. Furthermore, it is also known that context-aware approaches are effective in evaluating human activities if such evaluation is performed on data semantically structured in an ontology. Its effectiveness depends on how prior (i.e., domain) knowledge is used to structure data [31]. As shown in Section 2.3.3, many hybrid approaches are context-aware approaches.

The work done in this thesis extends context-aware architectures for HAR. Chapter 3 presents how Arianna+ uses its core components that are based on OWL-DL to reason on a set of ontologies interconnected in a network. Wherein, each ontology represents a certain specific context, e.g., an ontology could represent the context of an activity such as preparing a meal, and another ontology could represent the context of an activity such as watching TV. In chapter 5, sections 5.1 and 5.2, present systems based on Arianna+. The evaluation of these systems highlights that by having multiple ontologies - each with a dedicated reasoner – one can reduce the overall reasoning load compared to an approach wherein all relevant knowledge is encoded in a single ontology. Therefore, enabling a knowledge-based HAR system to be online in activity recognition and scalable in terms of accommodating more granular contexts or activities. This is especially relevant when each ontology is related to a detailed – but very partial, i.e., contextualized – representation of the application domain. Furthermore, chapter 5, section 5.4, presents a real-world HAR system based on Arianna+ that uses its online activity recognition capability to provide proactive support to the home user.

RQ3. How can data-driven and knowledge-based approaches be integrated and used together such that their results are accurate and intelligible?

As described in Section 2.3, on the one hand, data-driven methods are more flexible as compared to knowledge-based methods in terms of recognizing complex activities when there exists variability in the execution of sub-activities/actions. Furthermore, data-driven methods are robust against the intrinsic noise and uncertainty of sensor measurements. But they lack the capability of capturing important semantic relationships between sensor events and contexts or activities. Moreover, data-driven methods often need large annotated training

data. On the other hand, knowledge-based methods capture very well the above-mentioned complex semantic relationships. These methods use the domain knowledge to conceptually model an activity. In this way, an activity can be modeled without the need for large training data. But their specification is often too rigid to cope with the variability in execution of sub-activities/actions and to handle noise and uncertainty. Furthermore, knowledge-based methods may require extensive knowledge engineering effort.

Chapter 3 presents Arianna+ as a framework for developing a HAR system that can be architecturally hybrid. This in terms of being able to combine the strong points of both data-driven and knowledge-based approaches. Sections 3.3.3 and 3.3.4 highlight how data-driven and knowledge-based approaches work together for accurate and intelligible HAR. In chapter 5, sections 5.3 and 5.4, present real-world HAR systems based on Arianna+ that incorporate knowledge generated from data-driven methods to more accurately recognize composite activities.

RQ4. How can a HAR system accommodate heterogeneous sensors and process sensors' data and at the same time be intelligible in its results and inner-workings?

There are three major categories of data processing approaches, i.e., data-driven, knowledge-based and hybrid. As mentioned in the response of RQ3, Arianna+ can be architecturally hybrid. Hence it is able to combine the strong points of both data-driven and knowledge-based approaches. As it can accommodate different approaches for data processing – consequently it can accommodate heterogeneous sensors.

During the development of a HAR system, a developer may use APIs for processing sensors' data and/or may use pre-trained activity models for activity recognition. A developer programming in Java can easily integrate OWL ontologies into their architecture by using OWLOOP API . Chapter 4 presents OWLOOP. In chapter 5, sections 5.3 and 5.4, present real-world HAR systems - based on Arianna+ and programmed using OWLOOP API - that incorporate knowledge generated from heterogeneous sensors, while at the same time being intelligible in its results and inner workings.

RQ5. Activities may be executed sequentially, concurrently, or in an interleaved manner. How can a HAR system recognize activities executed in any manner?

As described in Section 2.1, complex (or composite) activities can be composed of a set of actions or activities. Depending on the duration and order of these sub-units, activities can be divided into sequential, concurrent, and interleaved activities.

In chapter 3, sections 3.3.3 and 3.3.4 present core components, namely, computational procedures, events, and conditions. Using these core components an arbitrary data processing technique can be concurrently interfaced with a predefined knowledge structure (i.e., ontology) to recognize an activity. Using Arianna+ framework knowledge in each ontology can be hierarchically structured and concurrently evaluated hence enabling recognition of sequential or concurrent or interleaved activities.

Chapter 3

Arianna+

Based on the motivations presented in Section 1.1.1 and taking into account all the research problems described in Section 2.4, this chapter presents Arianna+. The chapter is structured as follows. Section 3.1 gives a background in Description Logic (DL) and Ontology Web Language (OWL). Section 3.2 highlights the novelty and gives a description of Arianna+. Section 3.3 describes in detail the core components of Arianna+. Finally, Section 3.4 summarizes the chapter.

3.1 Preliminaries

3.1.1 Description Logic (DL) and Ontology Web Language (OWL)

One of the earliest definition of a computational ontology is that it is a specification of a conceptualization [104]. Hence, an ontology is a formal description of a domain of interest. The Web Ontology Language (OWL) is a semantic language standardized by the World Wide Web Consortium (W3C) [105]. It is used to represent knowledge about things, groups of things, and relations between things in a particular domain. An OWL ontology is a structured set of *axioms*, *i.e.*, symbolic statements that specify what is true in the domain of interest. OWL has three increasingly-expressive sub-languages, namely OWL-Lite, OWL-DL, and OWL-Full. Among them, the Description Logic (DL) formalism lies within the decidable fragment of the family of languages based on First-Order Logic, and it is one of most used OWL sub-language [106]. The languages in the OWL family use the open world assumption. Under the open world assumption, if a statement cannot be proven to be true with current knowledge, we cannot draw the conclusion that the statement is false.

An OWL-DL ontology represents knowledge using axioms based on *concepts* (also known as classes), *roles* (also known as relationships), and *instances* (also known as individuals). Hence an ontology is considered to be divided into three boxes, *i.e.*, a Terminological-box (T-box), a Role-box (R-box) and an Assertion-box (A-box). The T-box, R-box contains a hierarchy of classes and properties, respectively. The A-box contains instances of the classes in T-box.

Throughout this chapter, concepts are denoted by capitalized words, *e.g.*, ROOM, roles are denoted by camel-case words, *e.g.*, connectsWith, whereas instances are denoted by uppercase letters, *e.g.*, A or B, or words starting with an uppercase letter. Examples of axioms within an ontology include: (*i*) A:ROOM, that indicates classification of A as an instance of the concept ROOM, (*ii*) ROOM $\sqsubseteq$ LOCATION, indicating that all the instances of the concept ROOM are also instances of the concept LOCATION, (*iii*) (A,B):connectsWith, denoting a binary relation between the two instances A and B, and finally (*iv*) connectsWith.ROOM, which defines a concept containing each instance that connectsWith other instances of the concept ROOM. In the last example we can also include the cardinality restriction used to define *at least*, *at most*, or *exact* number of instances (*e.g.*, ≥ 2 connectWith.ROOM $\doteq$ CORRIDOR, or $=1$ has.DOOR $\doteq$ CABINET). Furthermore, axioms can have *conjunctions* $\sqcap$ and *disjunctions* $\sqcup$ between each other.

An ontology can be coupled with an OWL-based reasoner, such as Pellet, Hermit, etc., which can check for consistency among the axioms. The reasoners also derive facts (axioms) - making implicit knowledge explicit - by the reasoning mechanism [107] based on subsumption of concepts and instance checking. OWL-based reasoners can also process the Semantic Web Rule Language (SWRL) rules [108], such that it is possible to generate axioms based on the conjunction of other axioms.

Additionally, an OWL-based reasoner can process a query posed as incomplete axioms (*e.g.*, formalised with the SPARQL language), and can provide a solution involving sets of instances, roles or concepts that consistently complete the given incomplete axioms. A more thorough description of the functionalities of DL-based ontologies can be found in [77].

3.2 Arianna+

3.2.1 Novelty

Section 1.1.1 presents two motivations for the design of Arianna+. Firstly, to facilitate an iterative development and ease in collaboration between HAR system researchers/developers

and health-care domain experts in ADL. Secondly, to have robust HAR that can support smart-home applications. The first motivation demands two key requirements, i.e., modularity and intelligibility. The second motivation demands three key requirements, i.e., to be scalable, online and accurate.

To meet these requirements while taking into account all the research problems highlighted in Chapter 2, this chapter presents the design of Arianna+, a framework to develop networks of ontologies for knowledge representation and reasoning that enable smart homes to perform human activity recognition online. In the network, nodes are ontologies that represent different contexts and allow data contextualization, while edges are computational procedures that process data. The framework allows orchestration of data within a network of ontologies such that the orchestration results in contextualized HAR. Being a knowledge-based framework, Arianna+ is intrinsically intelligible. The framework allows integrating within the same architecture heterogeneous data processing techniques as computational procedures. Thus, with Arianna+, we do not propose a new algorithmic approach to HAR, instead, we present a modular knowledge-based architectural approach to accommodate knowledge-based and data-driven activity models in a context-oriented way. This is a novelty of Arianna+ as compared to other works in the literature.

3.2.2 Description

Arianna+ supports the design of smart home architectures for recognising ADL. It is based on a general-purpose definition of symbolic *statements*, which define the atomic knowledge that Arianna+ can process. Statements have an associated Boolean state and a timestamp. They can be classified in ontologies over time based on *a priori* knowledge as encoded in ontologies themselves. In Arianna+ , each relevant piece of knowledge is modelled as a statement. For instance, a statement can represent the fact that that a *cabinet's door is open*, whereas before *it was closed*, which might lead to different conclusions for different activities.

Arianna+ allows to design and maintain a network of ontologies, where *nodes* in the network are OWL ontologies, and *edges* therein are computational procedures. Each computational *procedure* implements an arbitrary algorithm that combines statements with the purpose of generating new statements. To this aim, we need to formalise a (communication) interface between computational procedures and ontologies. The more such an interface does not constrain the behaviour of algorithms encoded as computational procedures, the better is in view of an iterative development process. In particular, we assume that a computational

procedure implements an algorithm that (*i*) is triggered based on *events* occurring in an ontology, (*ii*) retrieves statements from given ontologies, (*iii*) performs a computable sequence of steps to aggregate knowledge in the form of statements, and (*iv*) generates statements – possibly – to be stored in other ontologies. If these conditions are met, we consider that the algorithm satisfies the knowledge *interface* of Arianna+ . It is noteworthy that Arianna+ by design does not distinguish *basic* statements generated directly using sensory data from *aggregated* statements generated by (a possibly long sequence of) computational procedures.

Arianna+ implements a scheduler to contextualise statements using the prior knowledge encoded in different ontologies of the network and triggering corresponding computational procedures. A detailed account of how the scheduler works is out of the scope of this chapter. However, it would suffice to say that its behaviour is based on how events are classified as statements in a given ontology of the network. A procedure is not only defined in terms of input and output statements, but also with the semantic associated with events characterising the particular context where it is computed, *i.e.*, an ontology. As a consequence, statements are subject to contextualisation, which in turn can generate events triggering further computational procedures.

Arianna+ is provided with an upper ontology defining ontologies and procedures that would be bootstrapped in the nodes and the edges of the network. In this paper, we present and discuss a network of ontologies able to spatially relate sensory data streams with the topological locations where the activities generating those data are supposed to be performed. Spatial information about the environment of interest is maintained within a purposely design ontology in the network. When an assisted person is in a given area or location, an event occurs and a procedure is scheduled to (*i*) select relevant spatially contextualised statements from such an ontology, (*ii*) use these statements to aggregate new, procedure-specific statements, and (*iii*) store the results in an activity ontology, *i.e.*, a temporal context. When a new statement is introduced in the temporal representation, an event might be detected and a procedure would be triggered to evaluate a *fluent model* and, eventually, recognise the activity.

Using this computational workflow, technology and domain experts can prototype, integrate, and evaluate heterogeneous techniques to activity modelling and recognition in a modular, flexible, intelligible and computationally efficient way.

3.3 Core components

3.3.1 Statements

We define a set $\mathbb{X}$ of statements, where each *statement* $X \in \mathbb{X}$ represents atomic knowledge about a context of interest. A statement X is characterised by a Boolean state $s_X \in \mathbb{B}$ observed at a time instant t_X. Therefore, a statement X can be defined as a tuple such that $X = \langle s_X, t_X \rangle$, where $t_X \in \mathbb{N}^+$ is a timestamp. In the notation we adopt, the name of a statement is always expressed in both the elements of the tuple, although we will discuss state and time independently if necessary. However, it is important to highlight that in Arianna+ one element of the tuple cannot exist without the other, since each statement is considered as atomic knowledge in each node of the ontology network.

We evaluate combinations of statements using a higher-order function $\boldsymbol{f}$, which is composed of a *logic* function $\boldsymbol{s}$ and an *algebraic* function $\boldsymbol{t}$. The function $\boldsymbol{f}$ maps a set of statements $\chi = \{X_1, \ldots, X_n\}$ to a new aggregated statement $\widetilde{Z}$. Please note that we always denote statements with Roman uppercase letters and functions with italic Roman lowercase letters. The higher-order functional $\boldsymbol{f}$ can be defined as

$$\boldsymbol{f} = \langle \boldsymbol{s}, \boldsymbol{t} \rangle \colon \begin{array}{rl} \mathbb{X}^n \to \mathbb{X} & \\ \chi \mapsto \widetilde{Z}, & \end{array} \quad \begin{cases} \boldsymbol{s} \colon & \mathbb{X}^n \to \mathbb{B} \\ & \chi \mapsto s_{\widetilde{Z}}, \\ \boldsymbol{t} \colon & \mathbb{X}^n \to \mathbb{N}^+ \\ & \chi \mapsto t_{\widetilde{Z}}. \end{cases} \tag{3.1}$$

In (3.1), $\boldsymbol{f}$ is treated as an *aggregator function*, where $\widetilde{Z}$ is a new statement having state $s_{\widetilde{Z}} = \boldsymbol{s}(\chi)$, and time instant $t_{\widetilde{Z}} = \boldsymbol{t}(\chi)$. In the special case when $n = 1$, we reduce the function to a single statement's definition, *i.e.*, $\boldsymbol{s}(X_1) = s_{X_1}$, and $\boldsymbol{t}(X_1) = t_{X_1}$, and therefore $\boldsymbol{f}(X_1) = X_1$.

In Arianna+ , the set of statements χ is considered to be the solution of a query to an OWL reasoner associated with an ontology in the network, *i.e.*, the ontology returns a set of statements contextualized with a common semantic (*e.g.*, all the statements related to the sensors attached to the refrigerator in a day period). The result of applying $\boldsymbol{f}$ is a statement in an ontology, which belongs to $\mathbb{X}$. In order to clearly distinguish raw statements from aggregated ones, we denote aggregated statements with a *tilde* symbol above them, as in $\widetilde{Z}$. We consider raw statements to originate from real physical sensors, and therefore a statement that is a result of $\boldsymbol{f}$ on a set of raw statements is considered to be an *aggregated* statement.

The function $\boldsymbol{f}$ aggregates statements based on the kind of operators used between them. As an example, assuming $n = 2$, *i.e.*, $\chi = \{X, Y\}$, we present the logical *and* operator in (3.2),

and the *precedence* operator in (3.3)

$$\widetilde{Z} \vDash X \wedge Y \iff \boldsymbol{f}: \begin{cases} \boldsymbol{s}: & s_{\widetilde{Z}} = s_X \wedge s_Y, \\ \boldsymbol{t}: & t_{\widetilde{Z}} = \max\{t_X, t_Y\}. \end{cases} \tag{3.2}$$

$$\widetilde{Z} \vDash X \leqslant Y \iff \boldsymbol{f}: \begin{cases} \boldsymbol{s}: & s_{\widetilde{Z}} = \top \iff t_X \leqslant t_Y, \\ \boldsymbol{t}: & t_{\widetilde{Z}} = \max\{t_X, t_Y\}. \end{cases} \tag{3.3}$$

In (3.2) and in the following discussion we use the symbols $\top$ to indicate *true*, and $\bot$ to denote *false*, as it is customary in DL-based formalisms. Given a set $\chi = \{X\}$, a Boolean state $\phi \in \mathbb{B}$, and a time instant $\delta \in \mathbb{N}^+$, we observe in (3.4) a specification of the logical *and* ($\wedge$), and a definition of the mathematical operator + for time computation purposes in (3.5):

$$\widetilde{Z} \vDash X \wedge \phi \iff \boldsymbol{f}: \begin{cases} \boldsymbol{s}: & s_{\widetilde{Z}} = s_X \wedge \phi, \\ \boldsymbol{t}: & t_{\widetilde{Z}} = t_X. \end{cases} \tag{3.4}$$

$$\widetilde{Z} \vDash X + \delta \iff \boldsymbol{f}: \begin{cases} \boldsymbol{s}: & s_{\widetilde{Z}} = s_X, \\ \boldsymbol{t}: & t_{\widetilde{Z}} = t_X + \delta. \end{cases} \tag{3.5}$$

It is noteworthy that it is possible to represent the logical *or* operator ($\vee$) similar to (3.2) and (3.4), whereas (3.3) can be used to define other comparison operators. Moreover, other mathematical operators like multiplication, division and subtraction can be defined as in (3.5). Other definitions of operators are legitimate (such as those inspired by the Allen's Interval Algebra), as long as they consistently provide a statement in $\mathbb{X}$ represented with a Boolean state $\boldsymbol{s}$, and a time instant $\boldsymbol{t}$, as shown in (3.1).

3.3.2 Fluent models

Using the formalism briefly drafted in the previous Section, we can design fluent models representing human activity. As a first example, let us consider an activity consisting of *picking two objects from a cabinet, using them for a while and placing them back in the cabinet*. In particular, let us consider a smart environment where those objects are associated with sensors able to generate statements I_4 and I_6 with state $\top$ if the objects are in the cabinet, and $\bot$ otherwise. Likewise, let us assume that a sensor is attached to the cabinet door D_7 with state $\top$ if it is open and $\bot$ otherwise. We define an activity model $\widetilde{A}_1$ that uses two aggregated statements, namely *object taken* $\widetilde{T}$, and *object released* $\widetilde{R}$, such that the activity is considered to be accomplished when those statements hold true after a minimum delay δ_1

between each other. The model is as follows:

$$\widetilde{A}_1 \vDash \left(\left(\widetilde{T} \wedge \top\right) + \delta_1\right) \leqslant \left(\widetilde{R} \wedge \top\right). \tag{3.6}$$

It must be noted that the model in (3.6) is represented as an aggregated statement $\widetilde{A}_1$, which holds true when the statement $\widetilde{R}$ becomes true after a δ_1 interval of time since $\widetilde{T}$ was true. For the sake of simplification, we denote using X^{ϕ} the result of the *and* operator applied to a statement X and a Boolean state ϕ of (3.4). Therefore, we may define

$$\widetilde{T} \vDash D_7^{\top} \leqslant \left(I_4^{\perp} \wedge I_6^{\perp}\right), \tag{3.7}$$

where the statements I_4 and I_6 indicate that the objects were not present after the cabinet door was opened. We also define

$$\widetilde{R} \vDash \left(I_4^{\top} \wedge I_6^{\top}\right) \leqslant D_7^{\perp}, \tag{3.8}$$

where the statements I_4 and I_6 indicate that the objects were present before the cabinet door was closed.

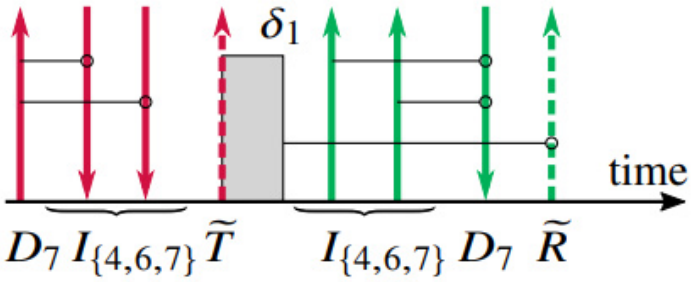


Figure 3.1 Graphical representation of the fluent model of activity $\widetilde{A}_1$.

More generally, the fluent models that can be designed in Arianna+ are fully compliant with the Allen's Interval Algebra, and it is possible to design complex activity recognition models composed of sub-models since $\boldsymbol{f}$ is a bijective function, which makes our representation modular and flexible. Although in this chapter we consider examples wherein statements are generated by simple Boolean sensors, it is nonetheless possible to obtain statements as a result of complex computational processes, for example, a classification of hand gestures recognised by neural networks or other methods that take inertial data from a wearable device. Nevertheless, in order to present the framework clearly, in this chapter we will rely on simple models like the one in the example introduced above. The fluent model

related to the example in (3.6) is shown in Figure 3.1, which uses a graphical formalism based on the statement's algebra.

As a second example, let us consider the problem of modelling an activity assumed to be accomplished if the person spends some amount of time in a specific location, *e.g.*, *cleaning*. Similarly to the previous example, let us consider the statement $D_{11}^{\top}$ when the door of the cabinet containing cleaning tools is open, and $D_{11}^{\perp}$ otherwise. We assume that presence sensors generate statements $\widetilde{L}$ and $\widetilde{K}$, when the person is located in the living room and in the kitchen, respectively. For this example, we can design a model that generates a $\top$ aggregated statement, when the door of the cabinet with cleaning tools has been opened, closed and, in the meanwhile, the person spent some time in the two rooms. For the fluent model of this example, we can define an operator for $\boldsymbol{f}$ that counts the number of statements in a given time interval. To this aim, the *convolution* operator $\circ$ counts statements $X_j \in \chi$ occurring within an time interval δ starting from the first statement in χ, such as

$$\left(\chi^{\phi} \circ \delta\right) = \left\{X_j : \forall j \in [1,m],\ s_{X_j} = \phi,\ t_{X_j} \in [t_0, t_0{+}\delta]\right\}, \tag{3.9}$$

where $t_0 = \min_j^m(t_{X_j})$ and m is the total number of statements in χ. Therefore, the aggregator function $\boldsymbol{f}$ using the convolution operator holds true when at least h elements are generated through convolution, *i.e.*,

$$\widetilde{Z} \vDash \chi^{\top} \circ_h \delta \iff \boldsymbol{f} : \begin{cases} \boldsymbol{s}: & s_{\widetilde{Z}} = \top \iff m \geqslant h, \\ \boldsymbol{t}: & t_{\widetilde{Z}} = \max_j^m(t_{X_j}). \end{cases} \tag{3.10}$$

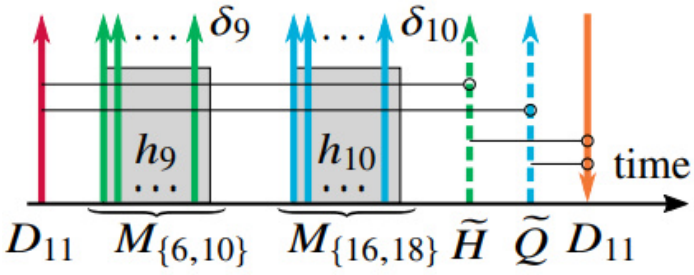


Figure 3.2 Graphical representation of the fluent model of activity $\widetilde{A}_7$.

Given a set of statements representing the location of the person in different rooms, *e.g.*, χ_L and χ_K, it is possible to define the cleaning activity fluent model as

$$\widetilde{A}_7 \vDash D_{11}^{\top} \leqslant \left(\left(\chi_L^{\top} \circ_{h_3} \delta_3\right) \wedge \left(\chi_R^{\top} \circ_{h_4} \delta_4\right)\right) \leqslant D_{11}^{\perp}, \tag{3.11}$$

wherein $h_3, h_4 \in \mathbb{N}^+$ are the minimum number of observations of the person in each room that are assumed to be required for the recognition of the cleaning activity, whereas $\delta_3, \delta_4 \in \mathbb{N}^+$ are the time intervals within which the minimum number of observations should take place, and all this should occur before placing back the cleaning tools into the cabinet and closing its door, *i.e.*, D_{11} becomes $\perp$. A graphical representation of this fluent model is shown in Figure 3.2.

3.3.3 Computational procedures

The network of ontologies is based on the DL formalism and is implemented using the OWL language. A statement $X \in \mathbb{X}$ represents an instance X of a concept $\Omega \sqsubseteq \mathsf{STATEMENT}$, *i.e.*, $\mathsf{X}{:}\Omega$. Moreover, each statement is associated with a Boolean state by a role, *i.e.*, $(\mathsf{X},s_X){:}\mathsf{hasState}$, and it is also associated with a time instant by a role, *i.e.*, $(\mathsf{X},t_X){:}\mathsf{hasTime}$. It is noteworthy that we specify only a subset of the roles the X instance must be involved in to logically describe it as an instance of $\mathsf{STATEMENT}$. However, X might also be described through other DL-based axioms that further specify it as an instance of a generic concept Ω, which can be used to retrieve contextualised statements. For instance, in the second example proposed in Section 3.3.2, a sensor might generate a set χ of statements X_i representing the person's location in the environment over time. In this case, each X_i might be classified as an instance of the $\mathsf{LIVINGROOM}$ or the $\mathsf{KITCHEN}$ concepts, and it would be possible to query an OWL-based reasoner to retrieve the person's location in a contextualised manner, *e.g.*, to obtain χ_L or χ_K, respectively.

Based on such a contextualised representation of statements, we can develop activity recognition models using a combination of $\boldsymbol{f}$ operators formalised through SWRL rules. In particular, we apply SWRL rules to select statements based on the context, and we use the logic function $\boldsymbol{s}$ as well as the algebraic function $\boldsymbol{t}$ to define fluent models for activities. However, SWRL-based models assume the *open-world* assumption and monotonic reasoning entailed by OWL. On the one hand, the open-world assumption does not allow to compute the maximum value of a set required by (3.2), (3.3) and (3.9), as the reasoner assumes that other unknown elements might exist. On the other hand, monotonic reasoning forbids the reasoner to generate new statements, but it allows to change the roles of existing statements, *i.e.*, change their Boolean state s_X and time instant t_X. Arianna+ relies on its computational procedures to overcome these limitations through an imperative programming language that the developer can use to design activity models based on a combination of logic-based and imperative-language based paradigms. Since it is not possible for an external procedure to

access and modify the knowledge in an ontology during the execution of the OWL-based reasoner, the framework synchronises the scheduling of procedures and the execution of the reasoning process.

SWRL Rules and Arianna+ Statements

A SWRL rule is an expression made by a conjunction of logic atoms, and if the expression is satisfied, then an implication is deduced.

$$a \wedge b \wedge c \implies d. \tag{3.12}$$

Therefore, if all the atoms $\{a, b, c\}$ are satisfied in an ontology, d is deduced, *i.e.*, the atom d will exist in the ontology.

SWRL atoms can represent the classification of an instance X in a concept of the ontology, *e.g.*, the DL expression X:$\mathsf{KITCHEN}$ is addressed as $\mathsf{KITCHEN}(\mathsf{X})$. Furthermore, an atom may be concerned with the properties beyween two instances X,Y, *e.g.*, the DL expression (X,Y):isIn is identified as $\mathsf{isIn}(\mathsf{X},\mathsf{Y})$. Atoms can be also used to perform basic algebraic computations and comparisons, and they support the definition of variables, which are indicated with the '?' symbol. To simplify the notation, we replace the atoms $\mathsf{greaterThan}(?\mathsf{g},\mathsf{k})$ with $(?\mathsf{g} \leq k)$, and $\mathsf{sum}(?\mathsf{r},\mathsf{k},?\mathsf{d})$ with $(?\mathsf{r} \leftarrow k + ?\mathsf{d})$, where the arrow identifies an assignment, and $k \in \mathbb{N}^+$ might be a variable.

Those types of atoms are evaluated by some OWL reasoner (*e.g.*, Pellet) to compute fluent models. In particular, SWRL rules are computed by checking all symbols' permutations that are consistent within the classes and properties encoded in an ontology.

3.3.4 Contextualized reasoning using multiple ontologies

Ontology Network

We represent all the knowledge relevant to a certain application domain by spreading it over a network of ontologies. More formally, we refer to such a network as a tuple $\mathbf{N} = \langle \omega, \rho \rangle$, where $\omega \supset \{\mathscr{O}_1, \ldots \mathscr{O}_i, \ldots, \mathscr{O}_d\}$ is a set of *at least* d nodes encoding ontologies, and $\rho \supset \{\mathcal{P}_1, \ldots \mathcal{P}_j, \ldots, \mathcal{P}_b\}$ is a set of *at least* b computational procedures. It is to be assumed that the nodes in the network are connected in any combinations among themselves through a set of computational procedures, which retrieve and provide statements from/to a combination of ontologies using high-order functions $\boldsymbol{f}$.

The number of ontologies and procedures in a network depends on the application, but the upper ontology $\mathscr{U} \in \omega$, and the scheduler $\mathcal{H} \in \rho$ (*i.e.*, a special type of procedure) are always involved in **N**. On the one hand, $\mathscr{U}$ must contain instances $\mathsf{O_i}$ and $\mathsf{P_j}$ that refer to each ontology $\mathscr{O}_i$ and procedure $\mathcal{P}_j$ in the network respectively. On the other hand, $\mathcal{H}$ gets bootstrapped automatically by Arianna+ , it loads in memory a node $\mathscr{O}_i$ for each ontology $\mathsf{O_i}$ in $\mathscr{U}$, and initialises the associated OWL-based reasoners. Afterwards, Arianna+ continuously observes the state of the $\mathscr{O}_i$ ontologies and, based on the events defining each $\mathsf{P_j}$ in $\mathscr{U}$, related $\mathcal{P}_j$ procedures are scheduled to pull, aggregate, and push statements for evaluating activity models.

The Upper Ontology

The upper ontology contains definitions pertaining to domain ontologies, procedures, and events.

In the network **N**, ω is the set of nodes that are initialised while bootstrapping an instance of Arianna+ . Each node $\mathscr{O}_i$ is represented with an instance $\mathsf{O_i}$ of the ONTOLOGY concept, that is defined in the upper ontology $\mathscr{U}$ as

$$\begin{aligned} =1\,\mathsf{represents.IRI} \sqcap \geqslant 1\,\mathsf{checkedBy.OWLREASONER} \\ \doteq \mathsf{ONTOLOGY}. \end{aligned} \tag{3.13}$$

The instance $\mathsf{O_i}$:ONTOLOGY is defined by the represents role, which relates $\mathsf{O_i}$ to the Internationalised Resource Identifier (IRI) of the ontology file containing DL axioms that define a context. Furthermore, (3.13) holds knowledge about a reasoner that processes the ontology based on OWL specifications for consistency checking, *e.g.*, $\mathsf{PELLET} \sqsubseteq \mathsf{OWLREASONER}$.

As discussed above, ρ in **N** is a set of procedures directly defined by *implementations*, which indicate how procedures themselves get computed. We simply refer to an implementation as an algorithm developed using some imperative language, which is associated with an activation event triggered on the basis of the context. Each procedure $\mathcal{P}_j$ exists in the network and it could be scheduled if a relative instance $\mathsf{P_j}$ is classified as an instance of the PROCEDURE concept, which is defined as follows in the upper ontology $\mathscr{U}$,

$$\begin{aligned} =1\,\mathsf{implements.IRI} \sqcap \geqslant 1\,\mathsf{requires.EVENT} \\ \doteq \mathsf{PROCEDURE}. \end{aligned} \tag{3.14}$$

In (3.14), the implements role associates an instance $\mathsf{P_j}$ with an identifier pointing to an algorithm, *i.e.*, the implementation of the procedure $\mathcal{P}_j$. The requires role, instead, associates

to $\mathsf{P_j}$ one or more instances E of the EVENT concept that the scheduler must check before activating the $\mathcal{P}_j$ procedure.

Events are defined in the upper ontology $\mathscr{U}$, and are associated with each PROCEDURE as shown in (3.14). Events represent situations when a statement X, within an ontology in ω, assumes a given state s_X. As a matter of fact, in $\mathscr{U}$, each instance E:EVENT is defined as a collection of conditions

$$\geqslant 1\,\mathsf{observes.CONDITION} \doteq \mathsf{EVENT}. \tag{3.15}$$

Each condition is in turn defined in $\mathscr{U}$ as an instance C of the concept CONDITION, defined as

$$\begin{aligned} =1\,\mathsf{checks.STATEMENT} \sqcap =1\,\mathsf{in.ONTOLOGY} \sqcap \\ =1\,\mathsf{hasTarget.BOOLEAN} \sqcap =1\,\mathsf{outcome.BOOLEAN} \sqcap \\ =1\,\mathsf{rate.HZ} \doteq \mathsf{CONDITION}. \end{aligned} \tag{3.16}$$

In (3.16), the checks role specifies a STATEMENT X to be evaluated in the ONTOLOGY $\mathscr{O}_i$, *e.g.*, (C,X):checks where X:STATEMENT and ($\mathsf{C,O_i}$):in where $\mathsf{O_i}$:ONTOLOGY. A C:CONDITION would have a $\top$ outcome if and only if the value specified through the hasTarget role is equivalent to the state s_X at certain instants of time, which are based on a rate specified in Hz.

Scheduling and Network Management

For each condition C available in the upper ontology $\mathscr{U}$, the system evaluates one or more logic rules at a given rate to classify C as one of the two disjoint concepts $\{\top\mathsf{CONDITION}, \bot\mathsf{CONDITION}\} \sqsubseteq \mathsf{CONDITION}$, depending on whether C's Boolean state outcome is $\top$ or $\bot$, respectively. Similarly, we use logic rules to define if an event E is an instance of $\top\mathsf{EVENT} \sqsubseteq \mathsf{EVENT}$. The event E is related to the condition C via the role (E,C):observes. Hence, every time C has a change in its outcome, the system applies the rules to classify E:$\top$EVENT when E is related to C:$\top$CONDITION only. When an instance E is classified in the $\top$EVENT concept, we assume it to be satisfied and, if it holds $\top$ that ($\mathsf{P_j}$,E):requires, then the scheduler runs the algorithm that such a $\mathsf{P_j}$ implements. It is noteworthy that, through appropriate combinations of conditions, we can specify logical *and* operators among checked outcomes, whereas via a combination of events we can specify logical *or* operators for triggering a procedure.

While a procedure performs computation to aggregate statements, conditions and events semantically identify the context in which such a procedure should be performed. In particu-

lar, conditions are evaluated through simple computations, *e.g.*, check the state of a given statement. Hence, the evaluation of an event (*i.e.*, a Boolean expression of conditions) is in qualitative terms *much simpler* than the execution of a procedure. As a consequence, the overall reasoning complexity is expected to be less as compared to the case where the corresponding procedures and contextualised representations are evaluated frequently in a single ontology.

Arianna+ relies on a generic definition of a statement, which can be used not only to represent the knowledge required by activity recognition models via procedures, but also to define the events activating those procedures. Therefore, all the ontologies in ω depend on the upper ontology $\mathscr{U}$. In particular, on the definition of the STATEMENT concepts (Section 3.3.1), which is shared among all the $\mathscr{O}_i$ ontologies. In addition, also the procedures in ρ depend on $\mathscr{U}$ since it specifies context-relevant algorithms, the input and output statements they concern, and activation events. Also, procedures can depend on other ontologies in ω because their algorithm may require access to certain context-specific statements at runtime.

An ontology network developed with Arianna+ undergoes two temporal phases, namely bootstrapping and running. The first phase involves the scheduler activation, which retrieves concept instances represented in the upper ontology $\mathscr{U}$, and generates three maps (shown in Figure 5.8). The first map contains all the ontologies $\mathscr{O}_i$, which are initialised in accordance with (3.13). The second map specifies the algorithm that each procedure $\mathscr{P}_j$ implements and their associated EVENT instances (3.14). The third map spans out a *periodic task* C_k with a specified rate for each C:CONDITION, which is initialised as $\bot$ in the upper ontology. Such a task is used to schedule procedures based on the outcome of each event, which is determined on the basis of all the C conditions that each event observes (3.15). In the second phase, the scheduler waits for notifications of condition changes from the periodic tasks. Based on these notifications, the scheduler eventually classifies some events as $\top$EVENT, and runs the relative procedures. When a procedure related to E is scheduled, E will not be an instance of $\top$EVENT anymore until consistent conditions re-occur. Finally, when $\mathscr{P}_j$ is active, the scheduler provides it with the map containing the ontology $\mathscr{O}_i$, and $\mathscr{P}_j$ begins to, pull, aggregate and push statements among ontologies in ω for evaluating activity models in a synchronised manner.

3.4 Summary

The motivations for designing Arianna+ are mentioned in Section 1.1.1. There are two motivations. Firstly, to facilitate an iterative development process and an enriched collaboration between HAR researchers/developers and health-care domain experts in ADL. Secondly, to have robust HAR that can support many smart-home applications with contextual knowledge. Hence, based on the first motivation, the key requirements of a HAR system are modularity and intelligibility. Furthermore, based on the second motivation, the key requirements of a HAR system are scalability, online HAR, and accurate HAR.

Hence, Arianna+ is a framework for HAR that adopts the nodes-and-edges design. Where the nodes are ontologies and edges are computational procedures. The ontologies are *a priori* defined contextual knowledge structures that can be updated with statements (i.e., axioms) based on sensor data. The computational procedures transfer or generate statements. A transfer of statements takes place from one ontology to another based on pre-defined events and conditions. New statements are generated by reasoning on the fluent models (i.e., rules) present in the ontologies. These statements are mapped from event-based sensor data. In order to exploit sensors providing a continuous data stream (e.g., inertial data or video) computational procedures can accommodate data-driven techniques, which can process the continuous data stream and provide an event as a result. Thus, in this manner, an ontology network is intrinsically *modular* due to its nodes-and-edges design and *intelligible* due to its knowledge-based approach.

The ability to accommodate both knowledge-based and data-driven techniques for the generation of statements (i.e., by reasoning on activity models) makes Arianna+ architecturally hybrid. Moreover, also *scalable* in terms of being able to accommodate more activity models within the network and data from heterogeneous sensors. This is because the activity models could be knowledge-based or data-driven hence they can deal with heterogeneous sensor data (i.e., event-based like data from PIR, pressure-mats, etc., or continuous-stream like data from accelerometers, gyroscopes, videos, etc.). Arianna+ is designed for *online* HAR. Although it is known that OWL-based reasoning is exponentially complex with respect to the number of axioms in the ontology [77], Arianna+ deals with this issue as it distributes knowledge over a network of ontologies and reasons only on contextually relevant knowledge. Arianna+ is designed for *accurate* HAR. Due to its modularity and scalability, one could design a redundant network, wherein multiple activity models (using different approaches) can be used to recognize the same activity and the most accurate HAR result can be selected based on a heuristic. A limitation of the Arianna+ framework is that the

reasoning is deterministic. The framework in its current state (i.e., with knowledge based on DL, and reasoning based on rules that are deterministic) is not robust to missing or noisy sensor data. This limitation can be overcome by further developing the framework to easily accommodate reasoning using fuzzy[81] or probabilistic [82] OWL reasoners.

Chapter 4

OWLOOP API

This chapter presents OWLOOP, an API that supports the development of HAR system architectures based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object- Oriented Programming (OOP). It addresses the research problems RQ1, RQ2, and RQ4 as described in Section 2.4. The chapter is structured as follows. Section 4.1 gives a background in Ontology Web Language (OWL) Application Programming Interface (API). Section 4.2 highlights the novelty and gives a description of OWLOOP API. Section 4.3 describes in detail the core components of OWLOOP API. Finally, Section 4.4 summarizes the chapter.

4.1 Preliminaries

4.1.1 Background

The Web Ontology Language (OWL) is a semantic language standardized by the World Wide Web Consortium (W3C) [105]. It is used to represent knowledge about things, groups of things, and relations between things in a particular domain. An OWL ontology is a structured set of *axioms*, *i.e.*, symbolic statements that specify what is true in the domain of interest, which can be *asserted*, *retrieved* and *inferred* by reasoning. The ability to reason on a semantic and dynamic knowledge representation (*e.g.*, involving relationships among sensory data, goal and actions) is particularly important for applications involving autonomous systems and human-machine interaction.

Ontologies are effective in many applications, which span in, but are not limited to, static representations (*e.g.*, to formalise the contents of a dataset [109] and the components of an architecture [110]), qualitative and quantitative models (*e.g.*, for biological [111] and

cognitive [112] systems), context-based systems (*e.g.*, to recognise human activity in a smart home [79] and for driving assistance [113]).

It is noteworthy that intelligent agents typically require a complex software architecture to support and implement their capabilities, *e.g.*, perceiving the surrounding environment, being aware of the context, understanding user intentions, taking decisions, and acting accordingly. In such scenario, an ontology should represent knowledge able to support each required functionality in a modular and flexible manner, since there could be different, alternative models and implementations for each component of the architecture that would be worthy of comparison and integration. Ontology-based data structures can help in the development of such architectures via a knowledge-centred design.

Moreover, a structured, formal and consistent representation of knowledge is expected to support incremental design and the thorough testing of individual system functionalities better than a more trivial data structure. If an ontology consistently defines each system functionality, it becomes progressively easier to integrate and compare different approaches to support the design of modular architectures, which can therefore be further integrated and extended. However, such a design is suitable only if the effort for developing a knowledge-centred architecture is lower than the one required to develop a simpler architecture using minimalist and *ad hoc* data structures for each functionality.

Conventionally, OWL-based API [114] and Jena [115] can be used to design and develop software that manipulates and queries ontology-based data structures, whereas the SPARQL Protocol and RDF Query Language [116] is a powerful query engine, and the Semantic Web Rule Language (SWRL) [117] can encode in the ontology queries having effects in the representation itself. Typically, OWL-based APIs support many deterministic reasoners, such as Pellet [118] and Hermit [119], as well as reasoners based on probabilistic inference [120] and fuzzy logic [121]. Several standard representations for different domains have been proposed and validated in the literature, with the purpose of being reused in different application scenarios [122]. Furthermore, specialised frameworks for particular domains have been developed (*e.g.*, for Robotics [123], Ambient Assisted Living [124], and Biology [125]), some of which allow representation learning in a data-driven fashion [126]. Last but not the least, graphical editors [127] and visualisers [128] are available.

4.2 OWLOOP API

4.2.1 Novelty

As described by the research problem in RQ2 of Section 2.4, often, the development and programming aspects, which are associated with the research in HAR systems, are neglected. They play an important role in an iterative development process [3], which is an integral part of collaborative work. Hence, chapter 4 presents OWLOOP, an API that supports the development of HAR system architectures based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object- Oriented Programming (OOP). The features of OWLOOP API are (*i*) to implement a general-purpose and modular OWL to OOP mapping, which can improve flexibility, reusability and maintainability of the HAR system, and (*ii*) to be performance-oriented since OWL reasoning is a resource-demanding task. To provide these features, OWLOOP defines flexible OOP interfaces that a developer can use to easily implement descriptors for a specific application. These interfaces enable a developer to choose the OWL axioms that are mapped into OOP objects and give the developer control of the reasoning task.

4.2.2 Description

From a software architecture perspective, an ontology is meant to be used in synergy with other software components [129]. Such components should be implemented through hierarchies of classes designed with the Object-Oriented Programming (OOP) paradigm, but issues occur since OWL axioms are not structured in an ontology following the OOP formalism. Although there are many similarities between the knowledge in an ontology and OOP's *classes*, *objects* and *properties*, there are also non-trivial differences, an exhaustive list of which can be found in [130]. This is one of the reasons why software developers are typically reluctant to use ontologies in their architectures [131].

As surveyed in [131], for accessing and integrating ontologies in an architecture, the active and passive *OWL to OOP mapping* can be used. The *active* mapping transforms an ontology from its syntactic form, *e.g.*, based on the Extensible Markup Language (XML), to code statements in a target programming language. With an active mapping, there is the possibility of reasoning over the executable ontology at runtime [132–134], but this process is limited to the amount of fitting between the OWL language and its OOP counterpart. Instead, with the *passive* OWL to OOP mapping [131], an ontology is integrated with the OOP language by the means of an external inference engine, *i.e.*, a reasoner like Hermit [119]

or Pellet [80], among others. The reasoner exploits the *factory pattern* to create immutable OOP objects containing snapshots of the knowledge in the ontology, which is an OWL-based data structure loaded in memory.

The passive mapping involves an *API-based* strategy, while the active mapping concerns an *ontology-oriented programming* strategy [131]. The passive mapping focuses on performance but it might lead to complex and voluminous source codes. Furthermore, with a passive mapping, OWL axioms that are related among each other in the ontology are always represented as independent OOP objects. Therefore, passive mapping allows to use of OWL axioms through OOP objects but it does not exploit the benefits of OOP paradigms as active mapping does. On the other hand, we are not aware of an active mapping that supports all the reasoning mechanisms implemented by the OWL reasoners used with passive mapping.

This chapter presents the OWLOOP API, which implements a passive OWL to OOP mapping for integrating ontologies and software architectures. Our objective is to entirely support OWL reasoners as well as exploiting the benefits of the OOP paradigm by avoiding the factory design pattern. The OWLOOP API maps OWL axioms into *descriptors*, which are OOP classes that represent a fragment of the ontology but they are not immutable and independent, *i.e.*, they can be designed within an OOP-based hierarchy of descriptors. The API's code metadata can be found in Table 4.1.

C1	Current code version	v2.1
C2	Permanent link to code/repository used for this code version	https://github.com/TheEngineRoom-UniGe/OWLOOP
C4	Legal Code License	GNU General Public License v3.0
C5	Code versioning system used	Git
C6	Software code languages, tools, and services used	Code language: Java
C7	Compilation requirements, operating environments & dependencies	Java v1.8.0, Gradle v5.2.1, Junit v4.12, aMOR v2.2, OWL-API v5.0.5, openllet v2.5.1
C8	Link to developer documentation/manual	github.com/TheEngineRoom-UniGe/OWLOOP/wiki

Table 4.1 Code metadata of OWLOOP API.

4.3 Core components

4.3.1 Definition of a Descriptor

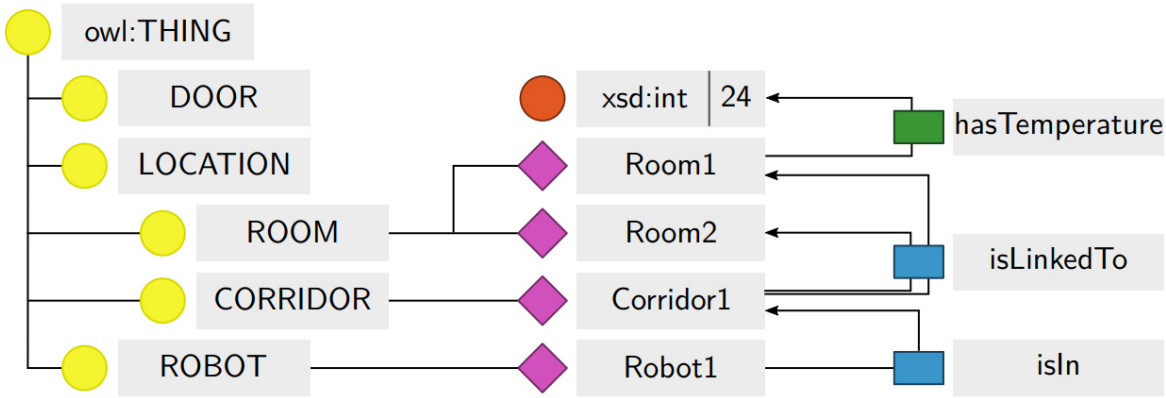


Figure 4.1 A simple ontology used as an example throughout Chapter 4.

Figure 4.1 depicts an ontology used as an illustrative example throughout this chapter. An OWL ontology is made of *axioms* $O = \{\mathcal{A}_1, \mathcal{A}_2, \ldots\}$, and each axiom is based on an *expression*[1] $\mathcal{E}$ relating some *entities*, *i.e.*, $\mathcal{A}_j = \mathcal{E}_k(e_1, e_2, \ldots)$. For instance, the illustrative ontology contains, among others, the OWL axioms

$$\begin{gathered} \mathsf{Super}(\mathsf{ROOM},\ \mathsf{LOCATION}), \\ \mathsf{Type}(\mathsf{ROOM},\ \mathsf{Room1}), \\ \mathsf{ObjectLink}(\mathsf{isLinkedTo},\ \mathsf{Room1},\ \mathsf{Corridor1}), \\ \mathsf{EquivalentRestriction}(\mathsf{CORRIDOR},\ \mathsf{min}\ 2,\ \mathsf{hasDoor},\ \mathsf{DOOR}). \end{gathered} \tag{4.1}$$

OWL defines $\mathcal{E}_k$ expressions with different semantics, and each of them is associated with OWL entities of specific types. Table 4.2 shows the types of entity, *i.e.*, an `OWLClass` (*e.g.*, ROOM) or an `OWLNamedIndividual` (*e.g.*, Room1), or an `OWLLiteral` (*e.g.*, 24), or an `OWLDataProperty` (*e.g.*, hasTemperature) or an `OWLObjectProperty` (*e.g.*, isLinkedTo). More details of the OWL formalisms are available in [135, 136].

The OWLOOP API enables interaction with OWL entities in an ontology by using *descriptors*, *i.e.*, Java-classes. We achieve a modular OWL to OOP map through descriptors that encapsulate reusable pieces of code required while accessing an ontology, *e.g.*, including

[1] OWLOOP provides an $\mathcal{E}$ counterpart for each OWL expression, *e.g.*, an OWL ObjectPropertyAssertion is an OWLOOP ObjectLink, while an OWL ClassAssertion is an OWLOOP Type. For simplicity, this chapter directly considers the OWLOOP expression shown in Table 4.3.

OWLEntity
OWLClass, OWLDataType, OWLNamedIndividual, OWLLiteral, OWLObjectProperty, OWLDataProperty.

OWLOOPEntity *(extends* OWLEntity*)*	
OWLOOPObject:	$\langle$OWLObjectProperty, OWLNamedIndividual$\rangle$,
OWLOOPData:	$\langle$OWLDataProperty, OWLLiteral$\rangle$,
OWLOOP Restriction:	$\langle$OWLClass$\rangle$, or $\langle C$, OWLClass$\rangle$, or $\langle C$, OWLObjectProperty, OWLClass$\rangle$, or $\langle C$, OWLDataProperty, OWLDataType$\rangle$.

Table 4.2 The definition of OWL and OWLOOP entities. The cardinality C concerns an OWL axiom among {some, only, min n, max n, exact n} where $n \in \mathbb{N}^+$.

textual constants and functions that usually lead to boilerplate code. This section presents the structure of OWL axioms in a descriptor, while Section 4.3.3 focuses on the descriptor functionalities to assert and retrieve knowledge subjected to reasoning.

The OWL to OOP mapping implemented by OWLOOP stores the axioms involving an $\mathcal{E}_k$ expression into a data structure $D_k = \langle x, Y_k \rangle$. In it, x is an OWL entity called *ground*, and $Y_k = k{:}[\{y_1\}, \{y_2\}, \ldots]$ is an *entity set* such that the OWL axiom $\mathcal{E}_k(x, y_i)$ is derived for each $y_i \in Y_k$. For example, we encode the OWL axioms in (4.1) as

$$\begin{aligned}
D_1 &= \Big\langle \mathsf{ROOM},\ \mathsf{Super}{:}\big[\{\mathsf{LOCATION}\},\ \{\mathsf{THING}\}\big] \Big\rangle, \\
D_2 &= \Big\langle \mathsf{ROOM},\ \mathsf{Instance}{:}\big[\{\mathsf{Room1}\},\ \{\mathsf{Room2}\}\big] \Big\rangle, \\
D_3 &= \Big\langle \mathsf{Room1},\ \mathsf{ObjectLink}{:}\big[\{\mathsf{isLinkedTo},\ \mathsf{Corridor1}\}\big] \Big\rangle, \\
D_4 &= \Big\langle \mathsf{Room1},\ \mathsf{DataLink}{:}\big[\{\mathsf{hasTemperature},\ 24\}\big] \Big\rangle, \\
D_5 &= \Big\langle \mathsf{CORRIDOR},\ \mathsf{EquivalentRestriction}{:}\big[\{\mathsf{min}\,2,\ \mathsf{hasDoor},\ \mathsf{DOOR}\}\big] \Big\rangle.
\end{aligned} \tag{4.2}$$

Each i-th element of the entity set Y_k is an OWLOOP entity, which can either be an OWL entity (as in D_1 and D_2) or a structure of OWL entities, *i.e.*, as specified in Table 4.2, an `OWLOOPObject` (in D_3), an `OWLOOPData` (in D_4) or an `OWLOOPRestriction` (in D_5). The latter is used to define OWL classes through a *cardinality*, *e.g.*, with a minimum number of hasDoor properties involving an individual classified as CORRIDOR.

OWLOOP represents the data structure D in the OOP interface named `Descriptor`, which is implemented as shown in Figure 4.3 through the Unified Modelling Language (UML), whose notation is summarised in Figure 4.2. From (4.2) it is possible to deduce that descriptors with the same type of ground might be merged to encode different axioms, *e.g.*, D_1 and D_2 can be represented in a *compound* descriptor $D_c = \langle x, Y_1, Y_2 \rangle$, where Y_1 and Y_2 concern the Super and Instance expressions. Hence, OWLOOP provides four different

Ground (x)	*Expression ($\mathcal{E}_k$)*	*Entity set element (y_i)*
`OWLClass` (`Class Descriptor`)	Equivalent	`OWLClass`
	Disjoint	`OWLClass`
	Super	`OWLClass`
	Sub	`OWLClass`
	Instance	`OWLNamedIndividual`
	EquivalentRestriction	`OWLOOPRestriction`
`OWLNamedIndividual` (`Individual Descriptor`)	Type	`OWLClass`
	Equivalent	`OWLNamedIndividual`
	Disjoint	`OWLNamedIndividual`
	ObjectLink	`OWLOOPObject`
	DataLink	`OWLOOPData`
`OWLObjectProprty` (`ObjectProperty Descriptor`)	Equivalent	`OWLObjectProperty`
	Disjoint	`OWLObjectProperty`
	Sub	`OWLObjectProperty`
	Super	`OWLObjectProperty`
	Inverse	`OWLObjectProperty`
	Domain	`OWLOOPRestriction`
	Range	`OWLOOPRestriction`
`OWLDataProperty` (`DataProperty Descriptor`)	Equivalent	`OWLDataProperty`
	Disjoint	`OWLDataProperty`
	Sub	`OWLDataProperty`
	Super	`OWLDataProperty`
	Domain	`OWLOOPRestriction`
	Range	`OWLOOPRestriction`

Table 4.3 The expressions which the current version of OWLOOP can map and their structure in descriptors.

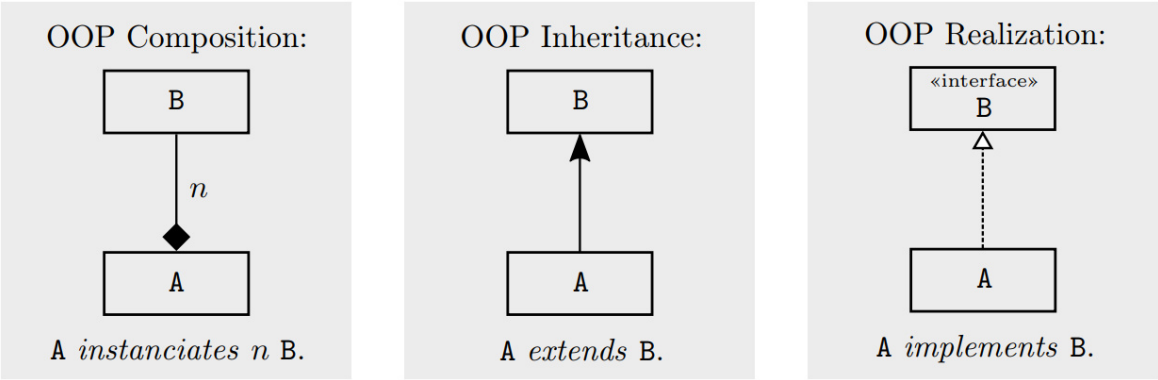


Figure 4.2 The UML notations used in Chapter 4.

descriptors based on their ground, *i.e.*, `ClassDescriptor`, the `IndividualDescriptor`, the `ObjectPropertyDescriptor`, and the `DataPropertyDescriptor`.

4.3.2 Construction of Descriptors

To address an application, developers should design suitable compound descriptors with sets of axioms based on Table 4.3. Four steps should be followed to design a compound descriptor based on the functionalities that an OOP class can inherit from OWLOOP interfaces. (*i*) Assign a ground entity by inheriting from one of the interfaces provided by the `*.descriptorGround` package. (*ii*) Inherit from extensions of the `Descriptor` interfaces (available in the package `*.descriptorExpression`) the representation of some expressions $\mathcal{E}_k$. (*iii*) Consistently for each $\mathcal{E}_k$ expression, instantiate an empty entity set based on the `*.descriptorEntitySet` package. (*iv*) For each $\mathcal{E}_k$, specify a descriptor to *build* (addressed in the next Section).

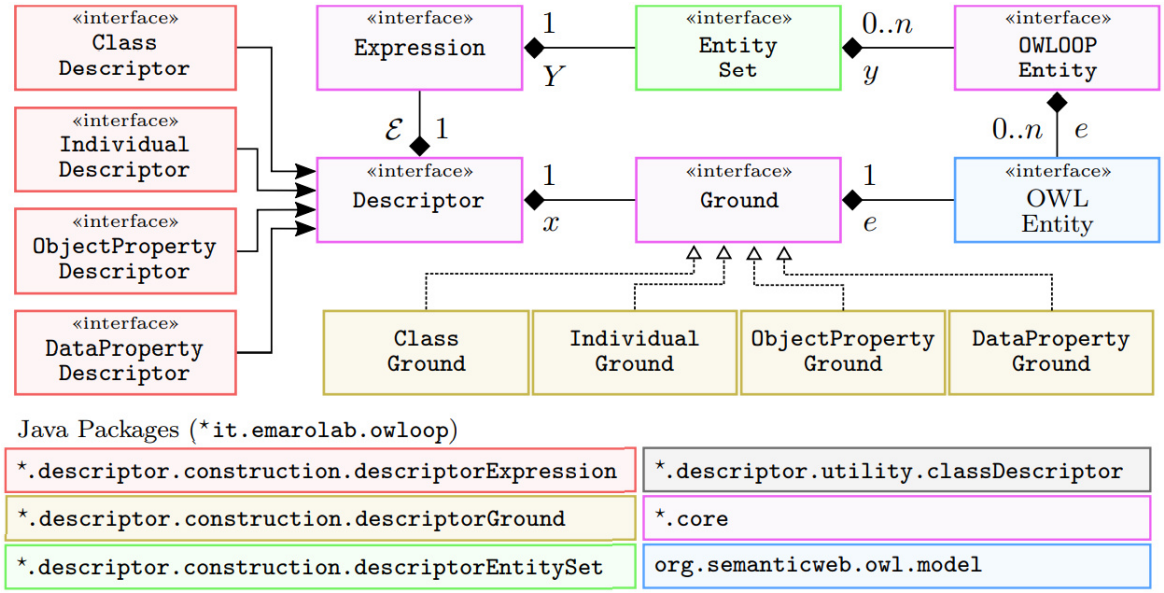


Figure 4.3 The general-purpose definition of OWLOOP `Descriptor`. Colours identify packages, *i.e.*, related classes, and they are also used in Figures 4.4–4.6.

Figure 4.4 shows the implementation of the `FullClassDescriptor`, which is a compound descriptor involving all the class expressions shown in the first row of Table 4.3. Therefore it requires as `EntitySet` four sets of `Classes`, a set of `Individuals` and a set of `Restrictions`. Remarkably, any OWLOOP compound descriptors with a `ClassGround` concerns a subset of expressions involved in the `FullClassDescriptor`. Figures 4.5 and 4.6 respectively show the implementation of the `FullIndividualDescriptor`, and the

`FullObjectPropertyDescriptor`, from which it is possible to derive the implementation of the `FullDataPropertyDescriptor`. Indeed, since our mapping exploits the same structure D for all OWL axioms, OWLOOP always relies on the same pattern to address different OWL expressions. Hence, OWLOOP is modular, and this also facilitates the implementation of the OWL axioms not considered in the current version.

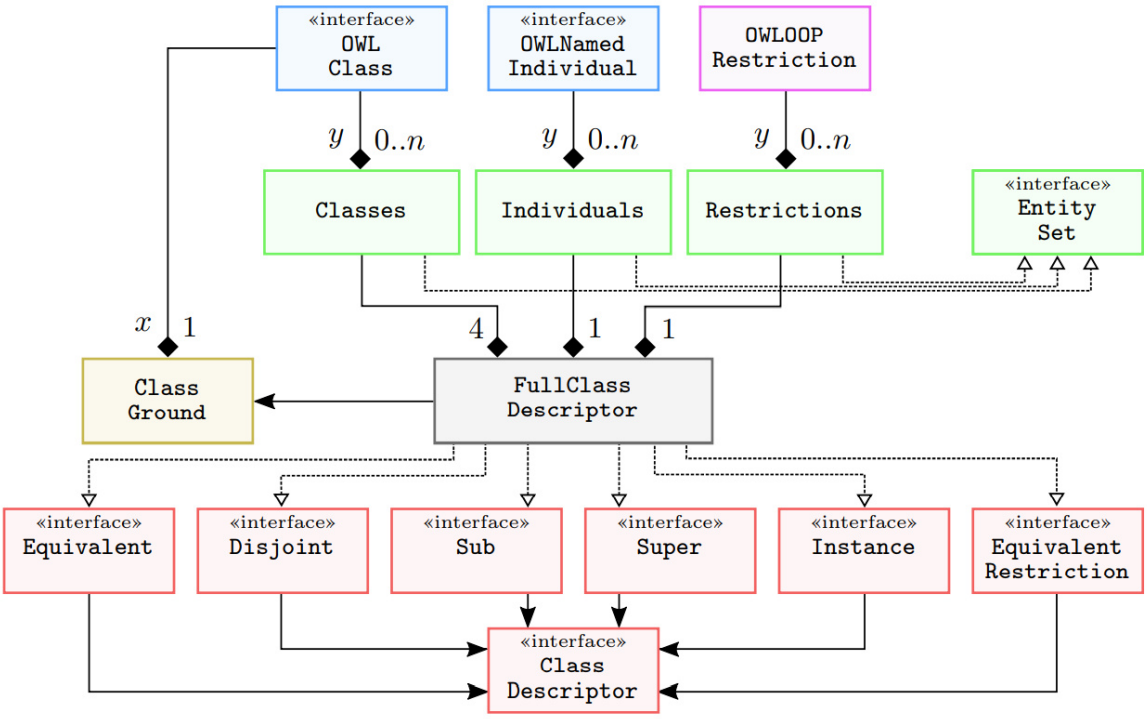


Figure 4.4 The implementation of the `FullClassDescriptor`.

4.3.3 Usage of Descriptors

Software functionalities

A descriptor derived from Table 4.3 can be instantiated given (*i*) a ground, *i.e.*, the identifier to an OWL entity (*e.g.*, "Corridor1"), and (*ii*) the reference to an ontology as required by aMOR[2], *i.e.*, an `OntologyReference`. For instance, Listing 1 shows at Line 1 a function to create the ontology with an associated reasoner, which should be manually invoked. As an example, Line 7 constructs a `LinkIndividualDesc`, which is a compound descriptor grounded on Corridor1 and is concerned with the expressions $\mathcal{E}$ inherited from the descriptors addressing ObjectLink and DataLink.

[2]OWLOOP is interfaced with the ontology through the aMOR library, which wraps the OWL-API and provides useful helper classes and functions.

```
1  private OntologyReference createEmptyOntology() {
2   OntologyReference.activateAMORlogging(false); // Disable logs.
3   return OntologyReference.newOWLReferencesCreatedWithPellet(
    "robotAtHomeOnto", // Ontology refeference name.
    "src/test/resources/robotAtHomeOntology.owl", // File path.
    "http://www.semanticweb.org/emaroLab/robotAtHomeOntology", // IRI.
    true // Synchronize the Pellet reasoner manually.
    );
4  }
5  private LinkIndividualDesc newCorridorDescriptorr() {
6   OWLReferences ontoRef = createEmptyOntology();
7   LinkIndividualDesc cd = LinkIndividualDesc("Corridor1", ontoRef);
8   return cd; // Compound descriptor concerning E:{ObjectLink, DataLink}.
9  }
```

Listing 1 Example showing the instantiation of an OWLOOP descriptor.

More generally, an OOP interface `D` that extends `Descriptor` for an expression $\mathcal{E}_k$ inherits the OOP *methods* addressed in the following paragraphs. These functionalities are based on the *internal state* of a descriptor (*i.e.*, the ground x and the entity set Y_k) that can be synchronised with the ontology. A compound descriptor is an OOP object `cd` that realises some `Descriptor` interfaces, whose functionalities can be accessed with `cd.`$\mathcal{E}_k$. For instance, the interfaces implemented by the object `cd` returned at Line 7 are accessible with `cd`.ObjectLink or `cd`.DataLink[3].

`D.getGround()`: This method returns the ground entity x. For a compound descriptor, `cd.`$\mathcal{E}_k$`.getGround()` returns the same entity for all the implemented expressions $\mathcal{E}_k$.

`D.getEntities()`: It returns the entity set Y_k associated to the $\mathcal{E}_k$ expression. To allow the definition of compound descriptors, each descriptor `D` defines this method with a different name, *e.g.*, `cd.getObjects()` for the ObjectLink expression, and `cd.getEquivalentIndividuals()` for Equivalent associated to an individual ground. These methods are used to access entities y_i, as well as manipulate them, *e.g.*, through the `cd.removeObject()` function.

`D.query()`: This method returns the knowledge related to the k-th expression that involves the ground entity in the ontology. It returns a set structured as Y_k but it does not affect the internal state of the descriptor. It is mainly used by the `Descriptor` interface itself.

`D.readAxioms()`: It relies on `D.getEntities()` and `D.query()` to compare the internal state of the descriptor with the state of the ontology. It changes the entity set of the descriptor such to be equal to the ontology. It returns a list of *intents* containing all the

[3] In this chapter, we slightly simplify the syntax. See the `*.articleExamples` package for the runnable version of the illustrated examples.

```
OWLReferences ontoRef = this.createEmptyOntology();
// Assert OWL Individual Expression Axioms in the ontology.
ObjectLinkIndividualDesc corrdor1 = // ℰ:{ObjectLink, DataLink}.
new ObjectLinkIndividualDesc("Corridor1", ontoRef);
corrdor1.addObject("isLinkedTo", "Room1"); // Add to ObjectLink EtitySet.
corrdor1.addObject(new OWLOOPObject("isLinkedTo", "Room2"));
corrdor1.writeAxioms(); // Synchronise changes to the ontology.
// Add OWL Class Expression Axioms to the ontology.
DisjointClassDesc robotClass = // ℰ:{Disjoint}.
new RestrictionClassDesc("ROBOT", ontoRef);
robotClass.addDisjointClass("LOCATION"); // Add to Disjoint EntitySet.
robotClass.addDisjointClass(ontoRef.getOWLClass("DOOR"));
corrdor1.writeAxioms(); // Synchronise changes to the ontology.
// Add OWL ObjectProperty Expression Axioms to the ontology.
DomainRangeObjectPropertyDesc hasDoor = // ℰ:{Domain, Range}.
new DomainRangeObjectPropertyDesc("hasDoor", ontoRef);
hasDoor.addDomainClassRestriction("LOCATION"); // Add to Domain EntitySet.
hasDoor.addRangeClassRestriction(new OWLOOPRestriction("DOOR"));
corrdor1.writeAxioms(); // Synchronise changes to the ontology.
// Synchronise changes to the entity sets based on the ontology.
ontoRef.synchroniseReasoner(); // Invoke OWL reasoning.
corrdor1.readAxioms();
hasDoor.readAxioms();
robotClass.readAxioms();
// Remove an axiom from the internal state of a descriptor.
corridor1.removeObject("isLinkedTo"));
corridor1.writeAxioms(); // Synchronise changes to the ontology.
```

Listing 2 Example showing how to manipulate axioms through descriptors.

performed changes, which can be used to recover from possible inconsistencies. A compound descriptor provides the method `cd.readAxioms()`, which invokes $\texttt{cd.}\mathcal{E}_k\texttt{.readAxioms()}$ for each k-th expression concerned by `cd`.

`D.writeAxioms()`: It is similar to `readAxioms()` but it changes the ontology to become equal to the entity set.

`D.build()`: It allows the use of a descriptor in an OOP manner since it returns a new compound descriptor `nd` grounded in each entity y_i. While designing compound descriptors, developers should specify the type of `nd` with a ground consistent with the elements in the entity set of `D`. Then, the entity set of the new descriptors is populated through the `nd.readAxioms()` method. Similarly to `D.getEntities()`, each descriptor `D` defines building methods with different names to allow the definition of compound descriptors. Remarkably, if entities y_i are not OWL entities, the `build()` method would have multiple definitions, *e.g.*, for an ObjectLink it is possible to return descriptors `nd` grounded on an object property or on an individual.

```
34  // Load the ontology shown in Figure 4.1 similarly to Line 1.
35  OWLReferences ontoRef = this.loadOntology();
36  // Retrieve the robot location (robotLoc) by assuming that it is unique.
37  LinkIndividualDesc robot1 = new LinkIndividualDesc("Robot1",ontoRef);
38  robot1.addObject("isIn", true); // Set to read only the isIn property once.
39  robot1.readAxioms();
40  OWLNamedIndividual robotLoc = robot1.getIndividualFromObject("isIn");
41  // Ground an individual descriptor on robotLoc concerning E:{Type}.
42  TypeIndividualDesc locIndiv = new TypeIndividualDesc(robotLoc, ontoRef);
44  locIndiv.readAxioms(); // read the classes of robotLoc.
46  // Build descriptors grounded on the classes of robotLoc.
48  Set<SubClassDesc> locClasses = locIndiv.buildTypes(); // E:{Sub}
50  for(SubClassDesc locClass : locClasses)
52  // A class of robotLoc that only subsumes owl:NOTHING is a leaf.
54  if(locClass.getdSubClasses().size() == 1)
56  // Print, e.g., "Robot1 is in Corridor1, which is a CORRIDOR".
58  System.out.println(robot1.getGround() + " is in " + robotLoc
    + ", which is a " + locClass.getGround());
```

Listing 3 Example showing the usage of the descriptor `build` method.

There are also other useful functionalities implemented by aMOR, *e.g.*, to get OWL entities or save the ontology. Moreover, the function `ontoRef.synchroniseReasoner()` performs OWL reasoning, which involves all the OWL expressions. Remarkably, OWL querying is a time consuming task if reasoning is required. Hence, the reasoning process should only be used when required, *i.e.*, to affect the results of `query()` and, consequently, the outcomes of `readAxioms()` and `WriteAxioms()`. In addition, to be compatible with other software using ontologies, aMOR provides access to factory-based OWL to OOP mapping implemented with OWL-API.

Illustrative examples

Listing 2 shows how to use compound descriptors to manipulate an ontology by creating an ontology with some of the axioms shown in Figure 4.1. At Line 13 the axiom ObjectLink(isLinkedTo, Corridor1, Room1) is added to the relative entity set, and the same operation is performed at Line 14 with a different type of input. At Line 15, the ontology changes such that Corridor1 is linked to Room1 and Room2. In addition, Lines 18–19 define axioms concerning the disjoint classes, and at Line 20, the ontology is changed to contain them, *i.e.*, Disjoint(ROBOT, LOCATION) and Disjoint(ROBOT, DOOR). Lines 23–24 define axioms Domain(hasDoor, LOCATION) and Range(hasDoor, DOOR), which are applied to the ontology at Line 25. Furthermore, Line 27 updates the reasoner to infer new knowledge in the ontology, which is then synchronised with the internal states of the descriptors at

Lines 28–30. Finally, Line 32 shows a way to remove an element from the internal state of a descriptor, and Line 33 applies those changes to the ontology.

The example in Listing 3 has the objective of finding the type of room where the robot is located, *e.g.*, CORRIDOR. The example shows the retrieval of the individual related to the isIn property, *e.g.*, Corridor1, that we store in a variable named `robotLoc`. Then, the example shows how to find the OWL classes that classifies `robotLoc` such that they are a leaf of the class tree in the ontology. Within Lines 37–40 the robot location is retrieved by assuming in the ontology that only an isIn axiom involving Robot1 exists. Lines 42–44 initialise a descriptor grounded on `robotLoc` that it is used to map OWL classes concerning the Type expression. Line 48 builds the Type of `robotLoc`, *i.e.*, it computes a set of descriptors that are grounded in a class representing the robot location (*e.g.*, CORRIDOR, LOCATION and owl:THING), and concerns Sub expressions. For each class of `robotLoc`, Line 54 checks whether it has only one subclass, *i.e.*, owl:NOTHING, which implies that it is a leaf class in the ontology. This example shows that OWLOOP allows for getting the OWL classes and subclasses in an OOP fashion, and a similar approach can also be used for the other expressions. Remarkably, Lines 40–44 show a less general implementation of the `build()` method. For instance, to retrieve `locIndiv` from `robot1` directly through building, the descriptor LinkIndividualDescr should be designed to build new TypeIndividualDesc objects.

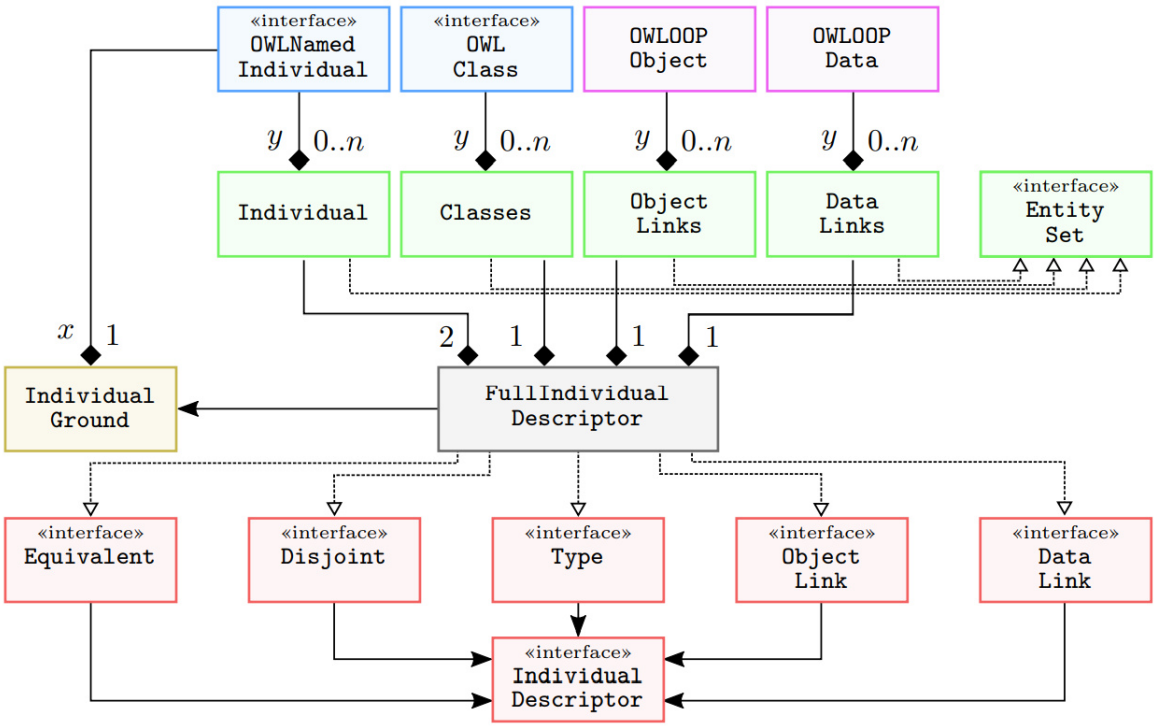


Figure 4.5 The implementation of the `FullIndividualDescriptor`.

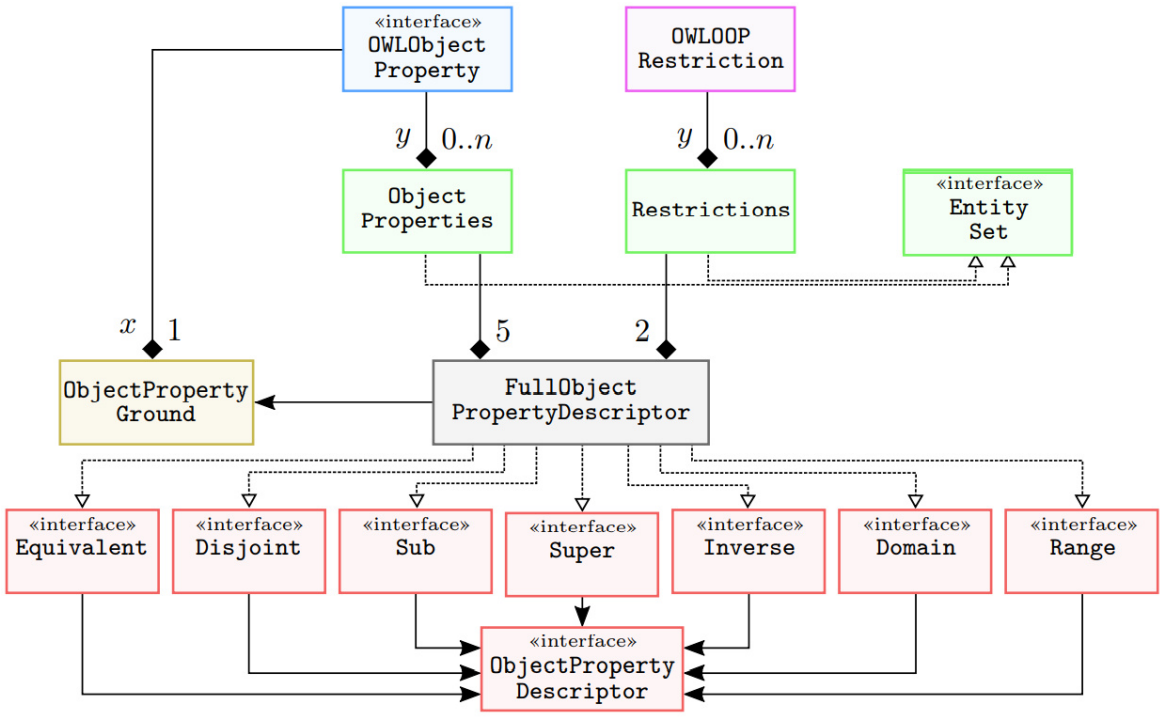


Figure 4.6 The implementation of the `FullObjectPropertyDescriptor`.

4.4 Summary

OWLOOP API is designed to support and ease the development of HAR systems based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object-Oriented Programming (OOP). OWLOOP API performs a passive OWL to OOP mapping that allows to manipulate knowledge in ontology files and perform inference by using OWL reasoners. OWLOOP allows to construct and use Descriptors, which are Java classes that interface OOP objects with knowledge structured in program-memory as an ontology. Descriptors encapsulate boilerplate code to simplify the development and maintenance of a system that exploits knowledge representation and reasoning using ontologies. The Descriptors' methods allow to read, write, update, delete, and reason on axioms in ontology files. Furthermore, flexibility in Descriptor construction allows to avoid drawbacks in computational performance. OWLOOP is suitable for complex applications requiring the management of dynamic ontologies.

The current version of OWLOOP concerns only with the expressions shown in Table 4.3, which do not encompass all the OWL axioms. For instance, it does not allow the representation of classes through structured disjunctions and conjunctions of class restrictions as

defined by OWL. Instead, it considers all the restrictions to be in conjunction with each other without a specific order. For most applications, this is not a limitation because it is possible to use an ontology editor (*e.g.*, Protégé [127]) to design static semantics. Then, the ontology can be loaded and subjected to runtime operations through OWLOOP descriptors. In the future, OWLOOP can be modularly extended to support all OWL axioms by designing new types of `OWLOOPEntity` and related `Descriptors`. Current limitations of OWLOOP are the following: (i) the effect on the computational performance (due to the overhead by the use of Descriptors, which are encapsulating Java-interfaces, classes, and methods in the OWL API) has not been quantitatively measured and (ii) based on the API design we hypothesize that flexibility in Descriptor construction allows to avoid drawbacks in computational performance - this also remains to be quantitatively confirmed. Nonetheless, these limitations can be overcome with further research into the API and they do not hinder the practical use of the API while building HAR systems - as will be seen in Chapter 5.

Chapter 5

Evaluating and exploiting Arianna+ using OWLOOP API

In this chapter, the following sections present HAR capable smart-home systems - as part of the work done in this thesis - that are developed based on Arinna+ (which was described with detail in Chapter 3) using OWLOOP API (which was described with detail in Chapter 4). Each section is divided into four subsections. The first subsection introduces the motivation, research problems the work resolves, and some background. The second subsection presents the methodology of experimentation. The third subsection presents the experimental setup, experiment, and results and evaluation. The final subsection summarizes the work and highlights how it addresses one or more of the research problems that were presented in Section 2.4. Furthermore, it highlights limitations (if any).

Moreover, on the one hand, the first two sections (5.1 and 5.2) present HAR systems that are evaluated in the laboratory using datasets. Work presented in Section 5.1 used a dataset based on a simple in-the-lab experimental-setup. Work presented in Section 5.2 used the publicly available CASAS dataset. On the other hand, the remaining two sections (5.3 and 5.4) present HAR systems evaluated with data from an experimental-setup at a volunteer's home. This evaluation at-home was the contingency plan. Originally, the plan was to collect a dataset from an assisted-living facility based on the work done by Teseo [1]. But the execution of this plan was not possible due to the Covid-19 pandemic [137, 138]. Hence instead, the contingency plan was executed to proceed with the thesis work.

[1] https://www.teseo.tech/

5.1 Evaluating computational performance of a network of ontologies

5.1.1 Introduction

This work presents a preliminary evaluation of a HAR system based on Arianna+ and developed using OWLOOP API. The evaluation showcases Arianna+'s scalability and online activity recognition. A positive evaluation of both these aspects highlights modularity of Arianna+. This work explores the research problems highlighted under RQ2 and RQ5, which are presented in Section 2.4.

Learning (or development) of activity models, in data-driven approaches, happens by training over datasets, whereas in knowledge-driven approaches it is done by explicitly encoding knowledge, typically in the form of a set of axioms, used for HAR based on sensor data. In terms of facilitating an iterative development process, the feasibility of the former approach is questionable since, if a new activity is to be introduced into the system, a new dataset has to be collected and the entire training process has to be performed. Whereas, the latter approach is more feasible as a new activity model's knowledge can simply be added as a set of axioms and rules.

In a real-world environment, robust HAR systems must guarantee scalability and online activity recognition. On the one hand, scalability can be achieve when (i) the system is *modular* with respect to activity models and (ii) types of sensors, as well as. On the other hand, online activity recognition depends on the *design* of the activity models.

This section presents a HAR system developed based on Arianna+. This system uses a hierarchy of ontologies, that decouple logic operations for semantically describing the context and support modular composition of reasoning behaviors for online activity recognition. The HAR system is presented from a software architecture perspective, and a use case is implemented, which is tested based on simulated data from distributed sensors.

5.1.2 Methodology

A network of ontologies

In Chapter 3, an ontology network is defined as a graph G, wherein the set of *nodes* N are ontologies (each with an independent DL reasoner) containing *statements* of the form (5.1),

i.e., having a Boolean state s and a generation timestamp t:

$$\texttt{Statement} \doteq \exists_{=1}\texttt{hasState}(s) \sqcap \exists_{=1}\texttt{hasTime}(t) \tag{5.1}$$

and are used to describe a specific part of the *context*, while the set of directed *edges* E are communication channels used for sharing statements between the nodes. Hence G is of the form:

$$G = \{N, E\} \tag{5.2}$$

where, $N = n_1, n_2, \ldots, n_n$, such that each node specializes in reasoning within a particular context, and $E = e_{12}, e_{13}, \ldots, e_{1n}, e_{21}, e_{23}, \ldots, e_{2n}, \ldots, e_{mn}$, such that the index of each edge signifies the direction of flow of statements, e.g., in e_{12} statements flow from n_1 to n_2. Consider an *event*, indicating that water is flowing from the sink in the kitchen. It can have different interpretations for a system aimed at recognizing activities such as cooking or cleaning. Instead of recognizing them actively from the same representation, with an ontology network it is possible to decouple their models in order to reason upon them based on an event or set of occurring events. Where, an event occurs based on rules that aggregate statements by logical conjunction. We show in the following Sections that this approach enforces system's modularity with respect to activity models, and if the network is such that it evaluates only the models related to a specific part of the overall context, then it also decreases the computation time.

The system checks the statements in the network with a given frequency and, when an event is detected, specific external *procedures* are executed in order to: (i) *move* statements from one node to another via edges, and (ii) evaluate models for activity recognition. For instance, statements could be generated from distributed sensors (e.g., detecting that Adam is in the kitchen at 8:00 am), then the system aggregates this information with prior knowledge to detect events (e.g., Adam is in the kitchen in the morning). When such an event occurs, the model for detecting that Adam is having breakfast gets evaluated by checking statements and their temporal relations within the model.

Moreover, activity models can generate statements, e.g., indicating that Adam had (or did not have) breakfast at a certain time, and hence can trigger new events, which can further be used to describe the context and evaluate models via procedure executions. A formal algebra of statements, used for defining events that execute procedures based on the context, has been proposed in 3.3.1.

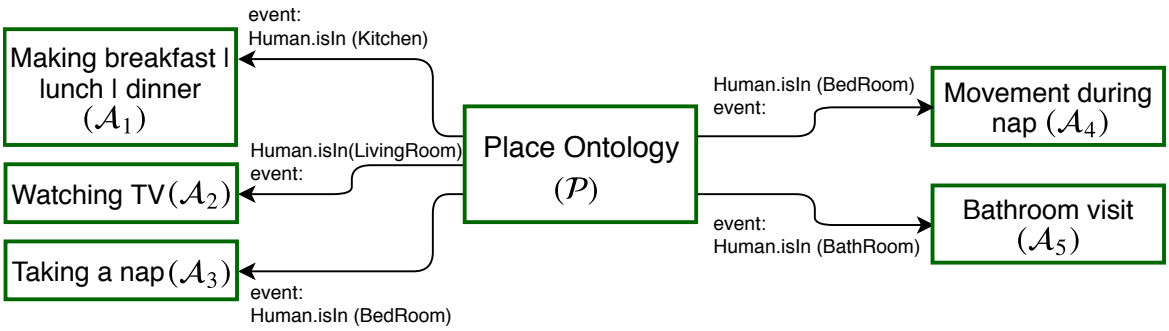


Figure 5.1 A simplified ontology network $\mathcal{O}$.

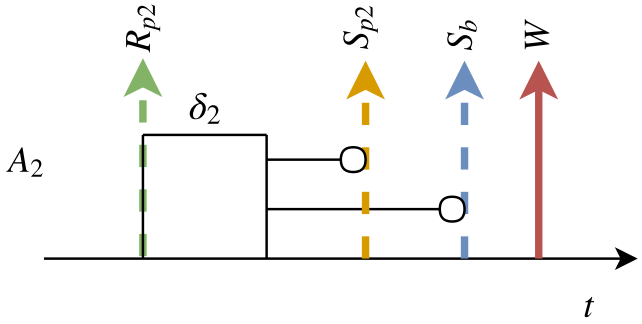


Figure 5.2 Visual representation of statements that make up the $\mathcal{A}_2$ model: statements are shown as vertical arrows where dashed arrows indicate information from $\mathcal{P}$, and solid arrows indicate statements generated by this model. Statement indexes indicate sensors influencing the state of that statement, while the temporal restrictions are shown as black lines.

Contextualized activity recognition

The ontology network developed in this work is represented as $\mathcal{O}$ as shown in Figure 5.1. In it there are 6 nodes; n_1 is a location-based contextualizing model called *Place Ontology* $\mathcal{P}$ and $n_2, \ldots, n_6$ are called *activity ontologies* $\mathcal{A}_i$, where $i = 1, \ldots, 5$ respectively. Nodes are designed such that statements within $\mathcal{P}$ take into account the spatial aspect, and statements within $\mathcal{A}_i$ take into account the spatial and temporal aspects of AR. $\mathcal{A}_i$ are listening for particular events that $\mathcal{P}$ generates, and the edges that link them are the following $E = e_{12}, e_{13}, e_{14}, e_{15}, e_{16}$. The nodes communicate and statements flow between them via edges, such that, $\mathcal{A}_i$ get activated and then evaluated by their independent reasoners, when a particular event occurs, as depicted by the graph in Figure 5.1. If the evaluation of an activity model gets satisfied, its procedure generates a new statement to notify the recognition of an activity, e.g., `WatchingTV.{hasState(True), hasTime(19:28)}`.

Within activity models, particular statements and temporal relations, must get satisfied for successful activity recognition. These are shown for $\mathcal{A}_2$, which recognizes the activity *WatchingTV*, in Figure 5.2. In it, statements are vertical arrows pointing upwards to indicate a *True* state and downwards for *False*. These statements are either transferred from another node (e.g., dashed arrows represent statements coming from $\mathcal{P}$), or are generated by this node (e.g., solid arrows are the statements generated by $\mathcal{A}_2$) and are indicated along with a name and an index or a range of indexes. A name is denoted by a capital letter and the sensors related to it are shown as the index. Statements are annotated along a relative x-axis, in order to restrict their temporal relations through black lines ending with a circle. In the Figure, we can see 4 statements: (i) statement R_{p2}, which is a dashed arrow of green color, is information coming from $\mathcal{P}$; it signifies `isIn_LivingRoom.{hasState(True), hasTime(19:25)}`, where the index $p2$ indicates that the sensor `PIR2` influences the state of this statement; (ii) statement S_{p2}, which is a dashed arrow of orange color, is information coming from $\mathcal{P}$; it signifies that there is some motion in the living room after δ_2 time units, naively representing the idea that, if Adam is sitting on the sofa then he is not sitting still; this statement can be replaced by a much robust statement, for instance, `sitting.{hasState(True), hasTime(19:26)}`, given that there may be other sensors in the system (e.g., wearable sensors, pressure sensors in the sofa); (iii) statement S_b, which is a dashed arrow of blue color, is information coming from $\mathcal{P}$; it signifies `highBrightnessTV.{hasState(True), hasTime(19:28)}`, where the indexes b indicates that `brightness sensor` influences the state of this statement; (iv) statement W, which is a solid arrow of red color, is generated when the overall model is satisfied, it signifies `WatchingTV.{hasState(True), hasTime(19:28)}`; this happens when statements S_{p2} and S_b are generated after δ_2 time units with respect to the R_{p2} statement.

In this system, when $\mathcal{A}_i$ receive statements from $\mathcal{P}$ the values of old instances get updated, if they are available. This has the effect of not accumulating statements in $\mathcal{A}_i$, i.e, the procedure related to it is in charge of updating and evaluating it, without accumulating time-related instances. Secondly, events are queries that return Boolean value when certain statements are satisfied, or not, in an ontology of the network. In the framework presented in Chapter 3, events are semantically defined in an upper-ontology that schedules related procedures if their query is verified. Whereas in this HAR system rather than having an upper-ontology, we have designed a system's architecture that incorporates the object-oriented programming (OOP) paradigm to execute $\mathcal{A}_i$ procedures with an *event-listener* pattern.

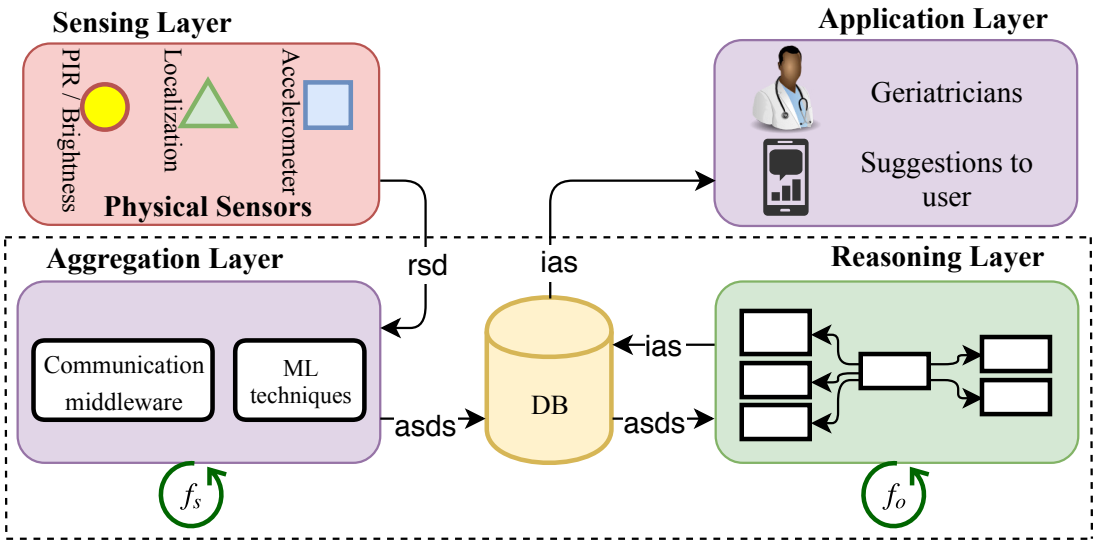


Figure 5.3 The system's architecture for the work presented in Section 5.1. Where, the link *rsd* signifies the flow of raw sensor data, *asds* signifies the flow of aggregated sensor data in the form of statements, *ias* signifies inferred activity statements and f signifies frequency.

5.1.3 Experimental evaluation

Experimental Setup

Figure 5.3 shows the system's architecture. It recognizes activities with $\mathcal{O}$ as described above; it comprises of the *sensing*, *aggregation*, *reasoning* and *application* layers. In this Section, we focus on the interfaces between those layers, which highlight the modularity aspect of Arianna+. Firstly, in the reasoning layer, $\mathcal{O}$ is used over time for recognizing activities based on data taken from the database (DB), which is getting accumulated with the latest sensor values and timestamps by the aggregation layer, which in turn is connected to the physical sensory layer. Finally, the application layer is used to easily interface geriatricians, other medical staff with the system's HAR results.

The *reasoning layer* is the system's core. It is made up of $\mathcal{O}$ and its internal working is as described in Section 5.1.2. There are two components in the working of this layer. The first is the initialization of $\mathcal{O}$ (i.e, TBox of ontologies are defined as nodes. While procedures and events are defined as edges). The second is the frequency f_o with which, in $\mathcal{O}$, the procedure of $\mathcal{P}$ takes in aggregated sensor data statements (link *asds*) from the database, updates the ABox, reasons (spatially) with knowledge within $\mathcal{P}$, and declares occurrence of an event, if any. If the declared event is being listened for by one or many $\mathcal{A}_i$, then their procedures get activated. Once an activity model's procedure is active, it takes in statements from $\mathcal{P}$ and

updates its own ABox, then reasons (spatially and temporally) with knowledge within the model and declares the recognition of a user activity. This completes a chain of reasoning processes (i.e., $\mathcal{P}$ *plus* an activity model), and if an activity is recognized in the process, then the procedure associated with the model saves the inferred activity statement (link *ias*) back in the database. As the reasoning process has not negligible computational time, if it is simply performed every time new sensor data statements arrive in the database, and if the frequency with which the new data arrives is faster than the reasoning process, then the system would not meet the near real-time constraint. Hence, we need f_o to have control over such a process. It deals with the computational complexity issue of the DL reasoner which performs the reasoning in $\mathcal{O}$.

From the *application layer*, on the one hand, geriatricians could visualize statistics related to the activities performed and explore further details in terms of statements (link *ias*), if necessary. On the other hand, the elderly individual could be stimulated with suggestions based on activity recognition, for instance, through dialogue-based interfaces via virtual coaches. Furthermore, the database also contains detailed logs of statements that were in $\mathcal{O}$, and therefore assistive or medical staff can access those statements to provide online services to the assisted individuals. For instance, a future scenario of in-home healthcare would be such that, if Adam is asked by his doctor about the number of times he visits the bathroom during the night, Adam's reply can be augmented by quantitative data from the smart home, which can help the doctor in making healthcare-related decisions.

The *aggregation layer* takes raw sensor data (link *rsd*) from heterogeneous sensors in the *sensing layer* and by using dedicated perception modules, processes the raw data to generate statements of the form (5.1). Then, it stores aggregated sensor data statements (link *asds*) in the database. This layer relies on a communication middleware module to channel all the Boolean data the sensors generate, and stores them in the database, if simple distributed sensors are considered. Furthermore, it relies on classification modules (e.g., obtained via machine learning approaches) that can provide statements with semantics (e.g., *sitting down*, *lying down*, etc), and stores them in the database, i.e, if sensors generating more complex data streams are considered. Remarkably, having a formal structure for a statement not only assures a modular evaluation of activity models, but also enables the overall AR system to take heterogeneous sensors into account. Statements are stored in the database at a frequency f_s, and moreover each perception module in this layer can have its own frequency at which it processes the raw sensor data to generate statements and store them in the database.

It is noteworthy that the frequencies f_s and f_o are independent of each other, such that, (i) the aggregation layer stores latest aggregated sensor data statements in the database at a

frequency f_s, which can be unique for different perception modules, and (ii) the reasoning layer reasons based on the latest statements that are available to it from the database, with a frequency f_o.

Experiment

The use case presented in this HAR system utilizes all $\mathcal{A}_i$ in $\mathcal{O}$, as shown in Figure 5.1. Their description is as follows. $\mathcal{A}_1$ infers *Making breakfast, lunch or dinner*. It is listening for the event ∃ `Human.isIn(Kitchen)`. It generates one of the statements, `Making breakfast` or `Making lunch` or `Making dinner`, when the assisted person uses furniture (e.g., the kitchen cabinet), after being present in the kitchen for a minimum time period of 60 seconds, and if that time period is inside one of the *a priori* defined intervals of the day, i.e., morning, afternoon or evening. $\mathcal{A}_2$ infers *Watching TV*. It is listening for the event ∃ `Human.isIn(LivingRoom)`. It generates the statement `Watching TV` when the occupant uses furniture (e.g., the TV), after being present in the living room for a minimum time period of 60 seconds, during any time of the day. $\mathcal{A}_3$ infers *Taking a nap in morning, afternoon or evening*. It is listening for the event ∃ `Human.isIn(BedRoom)`. It generates one of the statements `Taking a nap in morning` or `Taking a nap in afternoon` or `Taking a nap in evening`, when the assisted person uses furniture (e.g., the bed), after being present in the bedroom for a minimum time period of 60 seconds, and if that time period is inside one of the intervals of the day, i.e., morning, afternoon or evening. $\mathcal{A}_4$ infers *Movement during nap*. It is listening for the event ∃ `Human.isIn(BedRoom)`. It generates the statement `Movement during nap` when the person uses furniture (e.g., the bed) and the PIR associated with the bed remains active even after 60 seconds have passed on the bed, during any time of the day. $\mathcal{A}_5$ infers *Bathroom visit in morning, afternoon, evening or night*. It is listening for the event ∃ `Human.isIn(BathRoom)`. It generates one of the statements `Bathroom visit in morning` or `Bathroom visit in afternoon` or `Bathroom visit in evening` or `Bathroom visit in night`, when the assisted person uses furniture (e.g., the toilet seat), after being present in the bathroom for a minimum time period of 60 seconds, and if that time period is inside one of the intervals of the day, i.e., morning, afternoon, evening or night.

The use case is implemented by generating a simulated dataset with values and timestamps of a set of PIR sensors and a brightness sensor. It depicts a scenario where a person performs stereotypical activities that are held for eight minutes. The dataset is kept small so as to do extensive in-depth performance testing. The simulation is performed by updating the database with simulated sensor data in the form of statements (mimicking the link *asds*

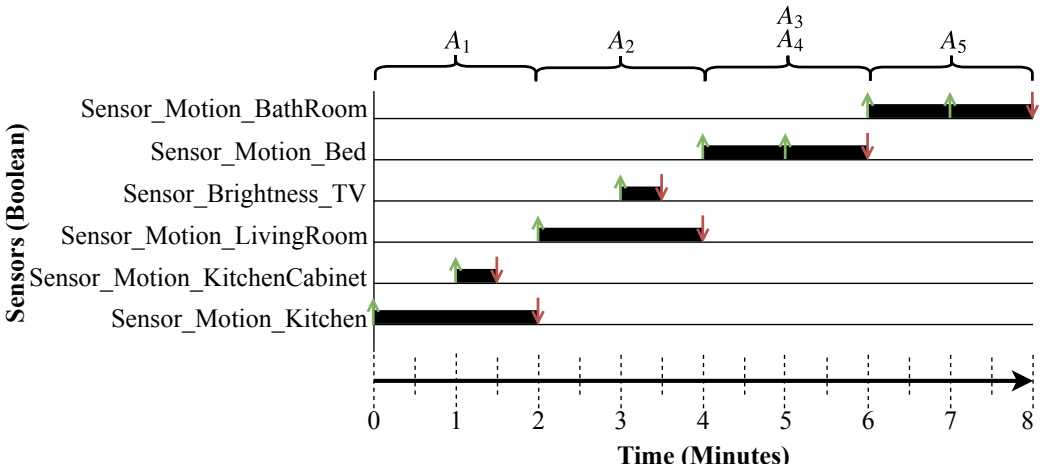


Figure 5.4 The simulated dataset used with the HAR system presented in Section 5.1.

connecting the aggregation layer and the database). As shown in Figure 5.4, Adam enters the kitchen, and after spending a minute in the kitchen, he opens the door of the kitchen cabinet and then closes it. He is in the kitchen for a total duration of 2 minutes. Next, he goes to the living room. After a minute in the living room, he switches on the TV and then switches it off after 30 seconds. He is in the living room for a total duration of 2 minutes. Next, he goes to the bedroom and simulates sleeping on the bed. He does not stay still in the bed, rather is constantly in motion. He is in the bedroom for a total duration of 2 minutes. Finally, the person goes to the bathroom. He is in the bathroom for a total duration of 2 minutes.

Among open source ontology reasoners that exist, e.g., Fact++, Pellet, Hermit and ELK. We use Pellet as it has more features in comparison [76] and is able to pinpoint the root contradiction or clash when inconsistency occurs. Experiments have been performed on a workstation with the following configuration: Intel® Core™ *i*7 2.6 GHz processor and 8 GB of memory. For assessing the system's performance, two types of evaluations are performed and compared. The first is the Contextualized Activity Evaluation (CAE), and the second is the Parallel Activity Evaluation (PAE). The CAE case represents the working of $\mathcal{O}$ as described in Section 5.1.2, where $\mathcal{P}$ behaves as a contextualizer such that an activity model gets activated based on the context. In the PAE case, $\mathcal{P}$ is no longer made to behave as a contextualizer, hence $\mathcal{A}_i$ are active in all contexts.

An evaluation (CAE or PAE) is performed as an experiment by setting a particular frequency f_o (of the reasoning layer). An experiment is performed with 5 iterations, with each iteration an extra activity model is added to $\mathcal{O}$ to increase the system's complexity. Each iteration is repeated 10 times to assess the reasoner's average computational time, and

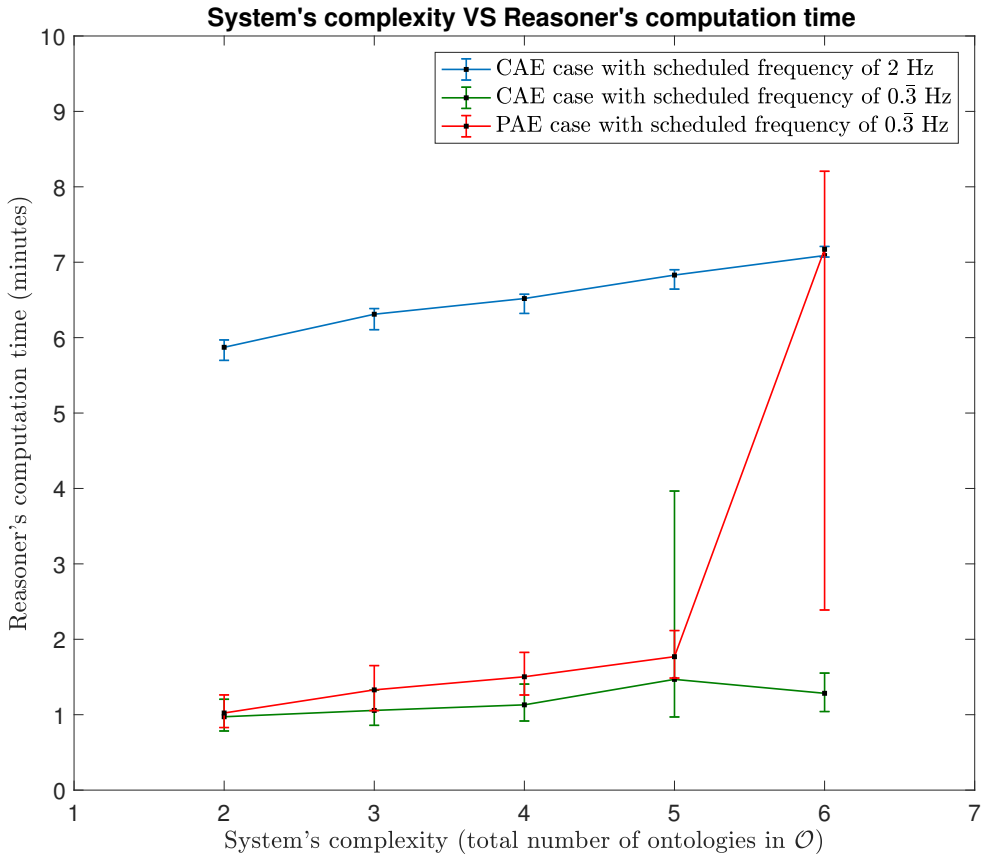


Figure 5.5 Ontology network's complexity *versus* reasoner's computation time. On x-axis are the number of ontologies, where 2 means $(\mathcal{P}+\mathcal{A}_1)$, 3 means $(\mathcal{P}+\mathcal{A}_1+\mathcal{A}_2)$, etc.

the maximum and minimum variance, from among 10 values. In total four experiments are performed, their process and results are described in the following Section.

Results

Performance results are shown in Figure 5.5, where x-axis shows the increasing number of ontologies in $\mathcal{O}$ (i.e., the number of activities the system attempts to recognize), with each iteration of an experiment. In relation to this, the y-axis shows the reasoner's computational time (i.e, the sum of the reasoning time spent in the ontologies of $\mathcal{O}$). A thread with a unique color represents a unique experiment conducted with a particular scheduled frequency f_o. A black dot on a thread marks the reasoner's average (10 repetitions of an iteration for an experiment) computational time, and vertical lines in the positive/negative direction (from a black dot) show the maximum/minimum variance, respectively, from the average

computational time. The simplest network has two ontologies, the Place Ontology $\mathcal{P}$ and $\mathcal{A}_1$, while the most complex network we tested has six ontologies, i.e., $\mathcal{P}$ and $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_5$.

Considering the CAE case, the reasoning layer is set to run with a time period of 500 milliseconds (i.e., f_o is 2 Hz), with the hypothesis that recognizing activities within 500 milliseconds is satisfying soft real-time constraint. Represented by the blue thread, the reasoner's computational time is high and increases linearly with the increase in system's complexity. Following the success of the previous test, and considering the PAE case, f_o is kept the same, i.e, 2 Hz. However, this case is not represented by any thread, as an undefined amount of time was being taken by the reasoner to finish the reasoning process. Following the drawback in the previous case, and considering the same case, i.e., PAE, the reasoning layer is set to run with a higher time period of 3000 milliseconds (i.e, f_o is $0.\bar{3}$ Hz), to make sure that the reasoning process completes within the frequency f_o, a condition which is satisfied with that time period. Represented by the red thread, the reasoner's computational time is initially low but then behaves exponentially, with the increase in system's complexity. Finally, following the success of the previous case, and considering the CAE case, f_o is kept the same, i.e, $0.\bar{3}$ Hz. Represented by the green thread, the reasoner's computational time is initially low and remains low, as it increases linearly with the increase in complexity of the system.

More in the discussion of the results:

1. Comparing the two CAE cases with frequencies 2 and $0.\bar{3}$ Hz, respectively, against each other, and against the PAE case with frequency at 2 Hz, we see that, with the approach described in Section 5.1.2 (i.e., represented by the CAE cases), it is possible to have activity recognition with a high frequency. This shows the system's ability to support near real-time applications.

2. In case of PAE, when there are 6 ontologies in O, a high variance is seen with the reasoner's average computational time on the higher end, thus confirming an exponential behavior. In case of CAE when there are 5 ontologies in O, a high variance is seen with the reasoner's average computational time on the lower end, thus confirming a linear behavior.

3. The PAE case (wherein, even if multiple smaller ontologies are used, all their reasoners are running in parallel), represented by the red thread; shows an evident exponential behavior and can be compared to using one large ontology in the system. It is known in the literature that with an increase in the number of axioms in an ontology the search space increases exponentially [77]. Therefore in comparison, the CAE case, with its

linear behavior, shows clearly the advantage of Arianna+'s modularity with respect to activity models and their contextualized evaluation. Furthermore, such claim is supported by the fact that our system does not accumulate instances within ontologies as it uses an external reasoner to deal with temporal aspects of reasoning (as described in Section 5.1.2) and stores the recognized activities in a database.

5.1.4 Summary

In this work, we present a HAR system based on Arianna+. The objective of this work is to perform a preliminary evaluation of the computational performance while reasoning using a network of ontologies. Hence, a system was developed whose core is a reasoning layer based on a network of ontologies. The network is grounded on activity models, statements, computational procedures, and events-listeners.

An experimental setup is designed with a use case scenario comprising of five activity models. Experiments were performed using a simulated dataset for evaluating the behavior of the network and its computational performance. Results with CAE (contextualized activity evaluation) indicate that a HAR system that performs contextualized-reasoning using a network of ontologies has better computational performance as compared to a HAR system that performs reasoning in a non-contextualized manner, i.e., in case of PAE (parallel activity evaluation), wherein all activities are evaluated together at the same time. As the complexity of the system increases (i.e., number of axioms in the system) the overall reasoning time is less for the case of CAE as compared to the case of PAE. However, we restrain from making conclusive statements about the comparison, because of the presence of high variability in the result of the CAE case (as can be seen in Figure 5.5).

Limitations of this evaluation are that it considers a single occupant in the environment and although tested extensively, a small and simulated dataset is used. Hence, future work could involve testing long-term computational performance with data arriving from a real-world smart-home setup. Furthermore, in the future, data-driven techniques that can process inertial data can be incorporated into computational procedures such that the network of ontologies can have sensor statements related to human actions and postures.

5.2 A HAR system based on Arianna+ evaluated using the CASAS dataset

5.2.1 Introduction

This work is an evaluation of a system based on Arianna+. The evaluation highlights scalability, online and accurate activity recognition. A positive evaluation of these aspects highlights modularity and robustness of Arianna+. Furthermore, the evaluation highlights intelligibility in the inner-workings of the system, i.e., intelligibility of activity models. This work explores the research problems highlighted under RQ1, RQ2 and RQ5, which are presented in Section 2.4.

This work shows that a system based on Arianna+ can be modular with respect to reasoning within multiple contexts, i.e., for different activities. It also shows that if each ontology in the network is concerned with a well-defined, small context, then the benefits are two-fold. (*i*) The knowledge *intelligibility* increases since each ontology addresses a limited and self-contained portion of knowledge that is affected by the reasoning process. Thus, domain experts can focus on specific aspects of the application, *e.g.*, modelling a given activity independently from the others. (*ii*) The reasoning load for data contextualisation is limited, and the architecture can be used *online*. This is because each ontology has a dedicated reasoner that evaluates only the data relevant for a certain context, *e.g.*, a particular activity. In addition, if the reasoners (and the other procedures) are invoked only when required by the means of well contextualised events, the computation would benefit and the flow of statements among ontologies would be of immediate comprehension.

For a scenario based on simple fluent models of ADL and a popular dataset involving event-based sensory data (presented in Section 5.2.3), this HAR system shows how Arianna+ can be flexibly used to develop an ontology network. Also, it reports the recognition and classification rates obtained through logically represented symbols, and compares it with other state of the art activity models validated with the same dataset. The comparison is done between systems that use a single representation domain (symbolic, probabilistic or hybrid) to encode the whole knowledge required to recognise human activities in the dataset, and an ontology network that involves distributed symbolic domains, *i.e.*, contexts. The comparison shows that on an ontology network can have recognition and classification performance that is comparable to state of the art techniques that are most often (to the best of our knowledge) based on a *one size fits all* approach to represent data

5.2.2 Methodology

Based on the timestamps available in the dataset, we simulate the streaming of data encoded as statements. The statements are used in an ontology network developed for the scenario described above and detailed in the next Section. We develop the network to show the design supported by Arianna+ and to assess its intelligibility and computational load.

The ontology network is designed to reduce the complexity of each context, *i.e.*, ontology. The results of such a design are discussed from three different perspectives in Section 5.2.3. (*i*) We observe the magnitude in reduction of the computational load. The computational load is a focus because logic-based reasoning is exponentially complex with respect to the amount of knowledge encoded in an ontology. (*ii*) We discuss the intelligibility level provided by Arianna+ through the examples. (*iii*) We analyse the differences between human activity recognition performance in an ontology network and in approaches exploiting a unique paradigm and/or data representation *corpus*, the main difference being that an ontology network allows to evaluate each activity in a purposely designed, specialised context.

With this experimental setup, we stress again that we do not want to compare our activity recognition models with other techniques validated in the literature, *e.g.*, upper ontologies for ADL [97] or MLN [139], but rather we want to emphasise the unique traits of the architectural aspects of Arianna+ . At the same time, we do not aim to investigate the recognition performance for specific activities, *e.g.*, meal preparation. Instead, we want to evaluate an ontology network in terms of its modularity, flexibility and intelligibility while it manages concurrent and interconnected contexts for recognising ADL.

Since Arianna+ enables an iterative development of human activity models, we showcase its modularity using a simple and engineered implementation of such models. In future Arianna+ iterations, we aim at comparing the recognition performance for specific activities and validating activity models. Such a validation phase might involve the adoption of computational procedures that are different from the ones used in this system. Furthermore, the validation phase may involve semantic representations – for describing ADL in ontologies – different from the spatial and temporal representations we propose here. Nonetheless, Arianna+ allows to flexibly integrate different procedures and ontologies since the only requirement is the proper use of *statements* (to represent data) and *events* (to trigger computation), as discussed in Sections 3.3.1–3.3.4.

5.2.3 Experimental evaluation

Experimental setup

We evaluate Arianna+ using the well-known CASAS dataset from Washington State University[2], as it is customary in ambient assisted living research. The dataset is presented in [140], and it contains data that was collected in an apartment with distributed sensors as shown in Figure 5.6. When a sensor generates Boolean information at some specific time instants, then statements are produced in the form of (3.1). The following sensors are present and considered in this scenario, (*i*) *motion sensors* generate $\boldsymbol{s}(M_i) = \top$ when the presence of a person is detected by the related sensor, (*ii*) *item presence sensors* generate $\boldsymbol{s}(I_i) = \bot$ when the related sensor detects that the item is absent, (*iii*) *flow detectors* generate $\boldsymbol{s}(F_i) = \top$ when water or gas is flowing, (*iv*) *door state detectors* generate $\boldsymbol{s}(D_i) = \top$ when the related door is open, and (*v*) *phone usage detector* generates $\boldsymbol{s}(P_i) = \top$ when the phone is being used.

Data have been acquired when residents in the apartment were performing a set of activities in an unscripted manner, and therefore data contain interruptions and concurrency in their execution. In total, 19 volunteers performed the following activities, which are enumerated as (α_1) filling the medication dispenser, (α_2) watching a DVD, (α_3) watering plants, (α_4) conversing on the phone, (α_5) writing a card, (α_6) preparing a meal, (α_7) cleaning the apartment, and (α_8) selecting an outfit from the wardrobe.

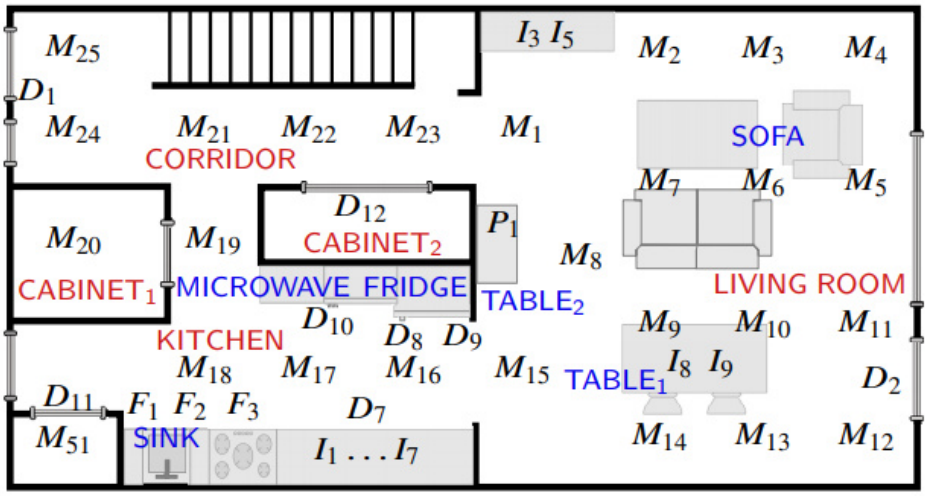


Figure 5.6 The sensors of a smart home, *i.e.*, motion detectors M_i, item-presence sensors I_i, flow sensors F_i, door sensors D_i, and phone P_i. Sensors are contextualized in a topological map concerning areas and furniture.

[2]Available at ailab.wsu.edu/casas/datasets/adlinterweave.zip

Configuring ontologies of the ontology network: Figure 5.7 shows relevant prior knowledge encoded inside the ontologies of the network, and it shows two disjoint sets of classes and sub-classes. The left-hand side concerns the localisation of a person in a room and their proximity to furniture. Instead, the right-hand side represents a hierarchy of classes concerning sensors and the type of activity they are supposed to monitor in a particular installation. If not explicitly stated, sub-classes are not disjoint, *e.g.*, we allow the presence sensor D_7:DOOR to be an instance of the MEDICINE and COOKING classes at the same time.

Figure 5.8 shows two types of ontologies, *i.e.*, the spatial ontology $\mathscr{L}$ and the activity ontology $\mathscr{T}_a$, which is instantiated for each activity $a \in [1,8]$. In the $\mathscr{L}$ ontology, we encode both hierarchies of classes shown in Figure 5.7, while each $\mathscr{T}_a$ shares with $\mathscr{L}$ only a part of the hierarchy on the right-hand side. In particular, $\mathscr{T}_a$ encodes only the part related to the statements provided by $\mathscr{I}_a$, as shown in Table 5.1 (and detailed further in the next Section). Remarkably, each ontology $\mathscr{T}_a$ also contains other prior knowledge, *e.g.*, an UPDATE concept to classify events for synchronising computational procedures, or a dedicated representation of TIME based on Allen's Algebra [64].

In $\mathscr{L}$, we spatially contextualise knowledge by defining

$$\begin{aligned}
\mathsf{PERSON} &\doteq \mathsf{isIn.LOCATION} \sqcap \mathsf{isNearTo.FURNITURE},\\
\mathsf{STATEMENT} &\doteq \mathsf{hasState}.\mathbb{B} \sqcap \mathsf{hasTime}.\mathbb{N}^+,\\
\mathsf{SENSOR} &\doteq \mathsf{STATEMENT} \sqcap \geqslant 1\,\mathsf{isIn.LOCATION}\\
&\qquad \sqcap\ \geqslant 0\,\mathsf{isNearTo.FURNITURE},\\
&\mathsf{LOCATION} \neg \mathsf{FURNITURE},\ \ \mathsf{SENSOR} \neg \mathsf{PERSON}.
\end{aligned} \tag{5.3}$$

Here, $\mathbb{B}$ represents the Boolean *concrete* concept, while $\mathbb{N}^+$ represents integer positive numbers in accordance with Section 3.3.1. Based on this representation, we consider an instance P:PERSON having properties (P,X):isIn and (P,Y):isNearTo and an instance S:SENSOR, which can have a high state, *i.e.*, (S,$\top$):hasState. In other words, given sensors with a $\top$ state, we spatially localise a person based on the properties (S,X):isIn and (S,Y):isNearTo.

The INSTALLED class describes sets of sensors that are related to a particular activity, and it is used to interface computational procedures as shown in Table 5.1. The sets of sensors are the following,

$$\begin{aligned}
\{\mathsf{D}_7, \mathsf{I}_4, \mathsf{I}_6, \mathsf{I}_7\}&\text{:MEDICINE},\\
\{\mathsf{I}_5, \mathsf{I}_3\}&\text{:TV},
\end{aligned}$$

$$\begin{aligned}
\{\mathsf{D}_{11},\, \mathsf{F}_2,\, \mathsf{F}_3,\, \mathsf{M}_6,\, \mathsf{M}_7,\, \ldots,\, \mathsf{M}_{14}\}&:\mathsf{WATERING},\\
\{\mathsf{I}_8,\, \mathsf{I}_9\}&:\mathsf{WRITING},\\
\{\mathsf{D}_8,\, \mathsf{D}_9,\, \mathsf{D}_{10},\, \mathsf{I}_1,\, \mathsf{I}_2,\, \mathsf{I}_7\}&:\mathsf{COOKING},\\
\{\mathsf{D}_{11},\, \mathsf{M}_6,\, \mathsf{M}_7,\, \ldots,\, \mathsf{M}_{10},\, \mathsf{M}_{16},\, \mathsf{M}_{17},\, \mathsf{M}_{18}\}&:\mathsf{CLEANING},\\
\{\mathsf{D}_{12},\, \mathsf{M}_3,\, \mathsf{M}_4,\, \ldots,\, \mathsf{M}_9,\, \mathsf{M}_{21},\, \mathsf{M}_{22},\, \mathsf{M}_{23}\}&:\mathsf{OUTFIT}. \qquad (5.4)
\end{aligned}$$

We denote SENSOR instances as D_i:DOOR, I_i:ITEM, M_i:MOTION, F_i:FLOW and H:PHONE, as shown in Figure 5.6. The activity of answering the phone is not listed in (5.4) because it is based on a single sensor, *i.e.*, H.

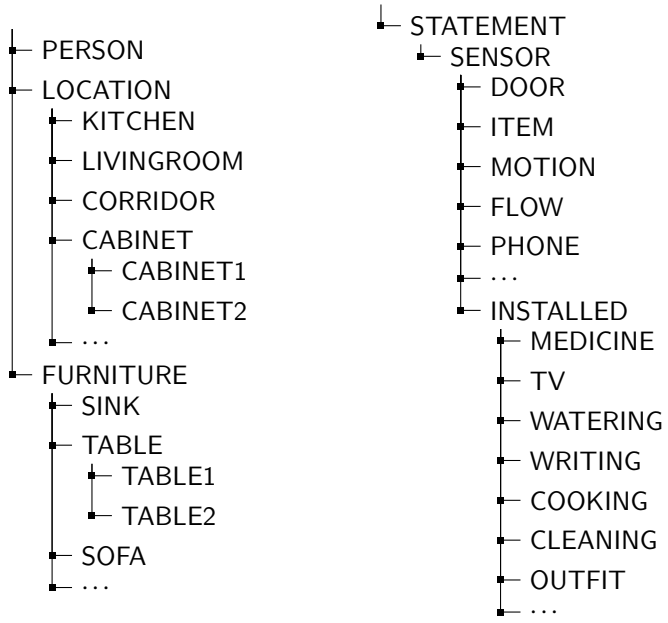


Figure 5.7 This figure partially shows the hierarchies of classes and sub-classes in the ontologies. They represent the prior knowledge used to address the scenario presented in Section 5.2.3.

Configuring computational procedures of the ontology network: The computational procedures are interfaced with the ontologies as shown in Table 5.1, *i.e.*, based on a scheduling event and given some required statements the computation of procedures are triggered, in accordance with Section 3.3.3.

Figure 5.8 shows three types of procedures. $\mathscr{D}$ creates instances of the SENSOR class in the $\mathscr{L}$ ontology, therefore simulating a stream of data. $\mathscr{D}$ updates the reasoner of the spatial ontology $\mathscr{L}$ each time it creates a new sensor statement in the ontology. Since $\mathscr{L}$ represents

Edge Name	Scheduling Events.	Required Statements.	Provided Statements.
$\mathcal{D}$	At bootstrap.	None.	X:SENSOR in $\mathscr{L}$.
$\mathcal{I}_1$	(P,K):isIn in $\mathscr{L}$, where K:KITCHEN, P:PERSON.	D_i:MEDICINE $\sqcap$ DOOR, I_i:MEDICINE $\sqcap$ ITEM in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_1$.
$\mathcal{I}_2$	(P,L):isIn in $\mathscr{L}$, where L:LIVINGROOM.	I_i:TV $\sqcap$ ITEM, in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_2$.
$\mathcal{I}_3$	(L,C):isNearTo $\vee$(P,S):isNearTo $\vee$(P,L):isIn in $\mathscr{L}$, where C:CABINET1, S:SINK.	D_i:WATERING $\sqcap$ CABINET1, M_i:WATERING $\sqcap$ MOTION, F_i:SINK in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_3$.
$\mathcal{I}_4$	(P,T2):isNearTo in $\mathscr{L}$, where T2:TABLE2.	H_i:PHONE in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_4$.
$\mathcal{I}_5$	(P,T1):isNearTo in $\mathscr{L}$, where T1:TABLE1.	I_i:WRITING $\sqcap$ ITEM in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_5$.
$\mathcal{I}_6$	(P,K):isIn in $\mathscr{L}$.	D_i:COOKING $\sqcap$ DOOR, I_i:COOKING $\sqcap$ ITEM in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_6$.
$\mathcal{I}_7$	(P,L):isIn $\vee$(P,K):isIn in $\mathscr{L}$.	D_i:CLEANING $\sqcap$ DOOR, M_i:CLAENING $\sqcap$ MOTION in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_7$.
$\mathcal{I}_8$	(P,R):isIn $\vee$(P,T1):isNearTo $\vee$(P,O):isNearTo in $\mathscr{L}$, where R:CORRIDOR, O:SOFA.	D_i:OUTFIT $\sqcap$ DOOR, M_i:OUTFIT $\sqcap$ MOTION in $\mathscr{L}$.	*the Required Statements and*, N:UPDATE in $\mathscr{T}_8$.
$\mathcal{M}_a$ $a\in[1,8]$	N:UPDATED in $\mathscr{T}_a$.	X:SENSOR in $\mathscr{T}_a$.	$\widetilde{A}_a$ in $\mathscr{T}_a$.

Table 5.1 The design of the interface among computational procedures and ontologies for the network shown in Figure 5.8.

a context that does not involve time, old statements generated by a sensor are overwritten when the sensors provide new statements.

$\mathcal{I}_a$ reacts to events related to the position of the person in accordance with Table 5.1. When $\mathcal{I}_a$ is scheduled, it moves statements that are relevant for the a-th activity from $\mathscr{L}$ to $\mathscr{T}_a$. The statements moved by $\mathcal{I}_a$ are retrieved by means of a sub-class of the INSTALLED concept that is relevant for a specific activity, as shown in (5.4) and Figure 5.7. $\mathcal{I}_a$ also creates a statement $\mathsf{N}^{\top}$ (in $\mathscr{T}_a$) with the purpose to generate an event that would trigger the computation of $\mathcal{M}_a$. In order to generate this event we encode in $\mathscr{T}_a$ prior knowledge such that N:UPDATE if $\boldsymbol{s}(\mathsf{N}) = \top$, while if it is $\bot$, N would not be an instance of the UPDATE class.

$\mathcal{M}_a$ is scheduled when in $\mathscr{T}_a$ it does exist a statement N classified as UPDATE. When $\mathcal{M}_a$ starts, it resets N, *i.e.*, it sets $\boldsymbol{s}(\mathsf{N}) = \bot$ in $\mathscr{T}_a$, and N is not be an instance of the UPDATE class anymore. Then, $\mathcal{M}_a$ computes the a-th activity model based on SENSOR statements in $\mathscr{T}_a$, which are collected over time. If the model recognises the activity, the aggregated statement $\widetilde{A}_a$ will be stored in $\mathscr{T}_a$, and all the old statements are removed from that ontology.

Since the models to recognise activities are based on SWRL rules (which are discussed in Section 3.3.3), $\mathcal{M}_a$ can update the reasoner of $\mathscr{T}_a$ to evaluate whether an activity is performed. However, SWRL rules do not support certain operations required to implement the statement's algebra, as discussed in Section 3.3.1. Therefore, we design $\mathcal{M}_a$ to perform some computation before evaluating the SWRL rule.

Configuring activity models in the ontology network: This section details the implementation of activity models shown in Figure 5.9. The models are specifically designed for the scenario presented in this Section and depicted in Figure 5.8.

Section 3.3.3 introduces SWRL rules, and it discusses some of their aspects related to Arianna+ . For each activity model shown in Figure 5.9, we provide (*i*) a formalisation with statement's algebra, (*ii*) an explanation with respect to prior knowledge, and (*iii*) an implementation based on logic and rules.

SWRL rules are applied to activity ontologies $\mathscr{T}_a$ that are not concerned with all the sensory data in the considered scenario. Instead, ontologies are concerned with only those sensor statements that are selected by $\mathcal{I}_a$ based on spatial contextualisation (Table 5.1). For clarity, in the models formalised with the statement's algebra, we explicitly indicate the sensor with respect to Figure 5.6, but this is not true in the notation of the SWRL rule. This is due to the fact that $\mathscr{T}_a$ represents a temporal context and not a spatial one, therefore, it does not encode the position of the sensors as it is done in $\mathscr{L}$. In other words, we assume that $\mathscr{T}_a$

encodes only sensor statements that are relevant for the a-th activity model. Consequently, there is the need to contextualise knowledge only in terms of sensors type and time intervals.

`Activity Model` α_1: The first activity concerns filling the medical dispenser. An implementation based on statement's algebra, and applied to the ontology $\mathscr{T}_1$, is[3]

$$\begin{aligned}
&\widetilde{A}_1 \vDash \big((\widetilde{T} \wedge \top) + \delta_1\big) \leqslant \big(\widetilde{R} \wedge \top\big), \text{ where} \\
&\quad \widetilde{T} \vDash D_7^{\top} \leqslant \big((I_4^{\perp} \wedge I_6^{\perp}) \vee (I_4^{\perp} \wedge I_7^{\perp}) \vee (I_6^{\perp} \wedge I_7^{\perp})\big), \\
&\quad \widetilde{R} \vDash \big((I_4^{\top} \wedge I_6^{\top}) \vee (I_4^{\top} \wedge I_7^{\top}) \vee (I_6^{\top} \wedge I_7^{\top})\big) \leqslant D_7^{\perp}.
\end{aligned} \tag{5.5}$$

The activity α_1 is recognised based on a statement $\widetilde{A}_1$ that can be expressed as

"two objects were taken ($\widetilde{T}$),
then after δ_1 time units, they were released ($\widetilde{R}$)."

Here, $\widetilde{T}$ and $\widetilde{R}$ can be expressed respectively as

"the door was opened, then the objects were absent.",
"the objects were present, then the door was closed."

The two *objects* are associated on the basis of prior knowledge, (5.4) and Table 5.1, *i.e.*, as

"medicinal items are in a cabinet of the kitchen."

The SWRL-based implementation of (5.5) is the following rule

$$\begin{aligned}
&\mathsf{DOOR}(?\mathsf{O}) \wedge \mathsf{hasState}(?\mathsf{O},\top) \wedge \mathsf{hasTime}(?\mathsf{O},\ ?\mathsf{t}_O) \wedge \\
&\mathsf{ITEM}(?\mathsf{I}) \wedge \mathsf{hasState}(?\mathsf{I},\perp) \wedge \mathsf{hasTime}(?\mathsf{I},\ ?\mathsf{t}_I) \wedge \\
&\mathsf{ITEM}(?\mathsf{J}) \wedge \mathsf{hasState}(?\mathsf{J},\perp) \wedge \mathsf{hasTime}(?\mathsf{J},\ ?\mathsf{t}_J) \wedge \\
&(\mathsf{I} \neq \mathsf{J}) \wedge (?\mathsf{t}_O \leqslant ?\mathsf{t}_I) \wedge (?\mathsf{t}_O \leqslant ?\mathsf{t}_J) \wedge \\
&\mathsf{DOOR}(?\mathsf{C}) \wedge \mathsf{hasState}(?\mathsf{C},\perp) \wedge \mathsf{hasTime}(?\mathsf{C},\ ?\mathsf{t}_C) \wedge \\
&\mathsf{ITEM}(?\mathsf{W}) \wedge \mathsf{hasState}(?\mathsf{W},\top) \wedge \mathsf{hasTime}(?\mathsf{W},\ ?\mathsf{t}_W) \wedge \\
&\mathsf{ITEM}(?\mathsf{Q}) \wedge \mathsf{hasState}(?\mathsf{Q},\top) \wedge \mathsf{hasTime}(?\mathsf{Q},\ ?\mathsf{t}_Q) \wedge \\
&(\mathsf{W} \neq \mathsf{Q}) \wedge (?\mathsf{t}_W \leqslant ?\mathsf{t}_C) \wedge (?\mathsf{t}_Q \leqslant ?\mathsf{t}_O) \wedge \\
&(?\mathsf{t} \leftarrow ?\mathsf{t}_O + \delta_1) \wedge (?\mathsf{t} \leqslant ?\mathsf{t}_C) \\
&\quad \Longrightarrow (\widetilde{A}_1,\top){:}\mathsf{hasState} \wedge (\widetilde{A}_1,?\mathsf{t}_C){:}\mathsf{hasTime}.
\end{aligned} \tag{5.6}$$

[3] In (3.7) we had considered a simplified case wherebe statements related to the object $I_7^{\perp}$ do not exist in the $\mathscr{T}_1$ context.

`Activity Model` α_2: The second activity is watching a DvD, and an implementation based on statement's algebra involving $\mathscr{T}_2$ is shown in (5.20). The statement $\widetilde{A}_2$, which represents the recognition of the α_2 activity, can be expressed as

"an object was taken, then after δ_2 time units, it was released,"

where the *object* is associated to prior knowledge concerning TV, ITEM, LIVINGROOM and related furnitures, *e.g.*, SOFA, in accordance with (5.4) and Table 5.1. The implementation of (5.20) based on SWRL is a rule

$$\begin{aligned}
&\mathsf{ITEM}(?\mathsf{T}) \wedge \mathsf{hasState}(?\mathsf{T},\bot) \wedge \mathsf{hasTime}(?\mathsf{T},\ ?\mathsf{t}_T) \wedge \\
&\mathsf{ITEM}(?\mathsf{R}) \wedge \mathsf{hasState}(?\mathsf{R},\top) \wedge \mathsf{hasTime}(?\mathsf{R},\ ?\mathsf{t}_R) \wedge \\
&(?\mathsf{t} \leftarrow ?\mathsf{t}_T + \delta_2) \wedge (?\mathsf{t} \leqslant ?\mathsf{t}_R) \\
&\quad \Longrightarrow (\widetilde{A}_2,\top):\mathsf{hasState} \wedge (\widetilde{A}_2,?\mathsf{t}_R):\mathsf{hasTime}.
\end{aligned} \tag{5.7}$$

`Activity Model` α_3: The third activity concerns watering plants, it involves $\mathscr{T}_3$, and (5.22) shows its definition with statement's algebra. The statement related to this model (*i.e.*, $\widetilde{A}_3$) can be expressed as

"the door was opened and the sink was used, then the person stayed near a plant for δ_3 time units, and near another plant for δ_4 time units, then the door got closed."

Here the furniture involved is represented with respect to prior knowledge representing a CABINET door, the SINK and two areas in the LIVINGROOM where the two plants are supposed to be. All those sensor statements are classified in $\mathscr{L}$ through the class WATERING shown in (5.4) and Table 5.1. With respect to Figure 5.7, we consider additional prior knowledge

$$\begin{aligned}
\{\mathsf{M}_6,\ldots,\mathsf{M}_9\}:\mathsf{PLANT1} &\sqsubseteq (\mathsf{WATERING} \sqcap \mathsf{MOTION}), \\
\{\mathsf{M}_{10}\ldots,\mathsf{M}_{14}\}:\mathsf{PLANT2} &\sqsubseteq (\mathsf{WATERING} \sqcap \mathsf{MOTION}).
\end{aligned} \tag{5.8}$$

The SWRL-based implementation of (5.22) is

$$\begin{aligned}
&\mathsf{DOOR}(?\mathsf{O}) \wedge \mathsf{hasState}(?\mathsf{O},\top) \wedge \mathsf{hasTime}(?\mathsf{O},\ ?\mathsf{t}_O) \wedge \\
&\mathsf{DOOR}(?\mathsf{C}) \wedge \mathsf{hasState}(?\mathsf{C},\bot) \wedge \mathsf{hasTime}(?\mathsf{C},\ ?\mathsf{t}_C) \wedge \\
&\mathsf{FLOW}(?\mathsf{F}) \wedge \mathsf{hasState}(?\mathsf{F},\top) \wedge \mathsf{hasTime}(?\mathsf{F},\ ?\mathsf{t}_F) \wedge
\end{aligned}$$

$$
\begin{aligned}
&(?\mathsf{t}_O \leqslant ?\mathsf{t}_C) \wedge (?\mathsf{t}_O \leqslant ?\mathsf{t}_F) \wedge \\
&\mathsf{WATERED}(?\widetilde{G}) \wedge \mathsf{hasState}(?\widetilde{G}, \top) \wedge \mathsf{hasTime}(?\widetilde{G},\ ?\mathsf{t}_{\widetilde{G}}) \wedge \\
&\mathsf{WATERED}(?\widetilde{E}) \wedge \mathsf{hasState}(?\widetilde{E}, \top) \wedge \mathsf{hasTime}(?\widetilde{E},\ ?\mathsf{t}_{\widetilde{E}}) \wedge \\
&(\widetilde{G} \neq \widetilde{E}) \wedge (?\mathsf{t}_F \leqslant ?\mathsf{t}_{\widetilde{G}}) \wedge (?\mathsf{t}_F \leqslant ?\mathsf{t}_{\widetilde{E}}) \wedge \\
&(?\mathsf{t}_{\widetilde{G}} \leqslant ?\mathsf{t}_C) \wedge (?\mathsf{t}_{\widetilde{E}} \leqslant ?\mathsf{t}_C) \\
&\quad \Longrightarrow (\widetilde{A}_3, \top):\mathsf{hasState} \wedge (\widetilde{A}_3, \mathsf{t}_C):\mathsf{hasTime}.
\end{aligned} \tag{5.9}
$$

Since SWRL respects the open-word assumption, we used further computation to perform the statements convolutions (3.9). This computation is performed by $\mathcal{M}_3$ before applying the above rule. In particular, $\mathcal{M}_3$ queries all the timestamps associated with the statements M_i:PLANT1$\sqsubseteq$SENSOR in $\mathscr{T}_3$, which contain the visits to locations of the plants; as imported by $\mathcal{I}_a$. Then, $\mathcal{M}_3$ computes the minimum (t^-) and maximum (t^+) values of the queried statements, and it counts their number (n). If $t^- + \delta_3 \leqslant t^+$ and $n \leqslant h_3$, then $\mathcal{M}_3$ generates a statement $\widetilde{G}$ with a $\top$ state and a timestamp equal to t^+. With an equivalent approach based on M_i:PLANT2 but with different parameters δ_4 and h_4, $\mathcal{M}_3$ also generates a statement $\widetilde{E}$. $\mathcal{M}_3$ generates $\widetilde{G}$ and $\widetilde{E}$ in $\mathscr{T}_3$ as instances of the WATERED$\sqsubseteq$WATERING$\sqsubseteq$SENSOR class, which represents that some water has been given to a plant. Then, $\mathcal{M}_3$ computes the activity model by updating the reasoner of $\mathscr{T}_3$, which evaluates (5.9).

`Activity Model` α_4: The fourth activity is answering to the Phone, which we define with statement's algebra in $\mathscr{T}_4$ as

$$
\widetilde{A}_4 \vDash \left(H_1^{\top} + \delta_5\right) \leqslant H_1^{\perp}. \tag{5.10}
$$

In $\widetilde{A}_4$, the object is associated with prior knowledge concerning H:PHONE. The SWRL-based implementation of (5.10) is the rule

$$
\begin{aligned}
&\mathsf{PHONE}(?\mathsf{T}) \wedge \mathsf{hasState}(?\mathsf{T}, \top) \wedge \mathsf{hasTime}(?\mathsf{T},\ ?\mathsf{t}_T) \wedge \\
&\mathsf{PHONE}(?\mathsf{R}) \wedge \mathsf{hasState}(?\mathsf{R}, \perp) \wedge \mathsf{hasTime}(?\mathsf{R},\ ?\mathsf{t}_R) \wedge \\
&(?\mathsf{t} \leftarrow ?\mathsf{t}_T + \delta_5) \wedge (?\mathsf{t} \leqslant ?\mathsf{t}_R) \\
&\quad \Longrightarrow (\widetilde{A}_4, \top):\mathsf{hasState} \wedge (\widetilde{A}_4, \mathsf{t}_R):\mathsf{hasTime}.
\end{aligned} \tag{5.11}
$$

`Activity Model` α_5: The fifth activity concerns writing a card, it involves $\mathscr{T}_5$, and (5.21) shows the related model based in statement's algebra, which outcome (*i.e.*, $\widetilde{A}_5$) can be expressed as

"two objects were taken,
then after δ_6 and δ_7 time units, they were released."

Here, the two *objects* are associated with prior knowledge that spatially describes WRITING in $\mathscr{L}$, as shown in (5.4) and Table 5.1. The implementation of (5.21) based on SWRL is the rule

$$\begin{aligned}
&\mathsf{ITEM}(?\mathsf{I}) \wedge \mathsf{hasState}(?\mathsf{I},\bot) \wedge \mathsf{hasTime}(?\mathsf{I},\ ?\mathsf{t}_I) \wedge \\
&\mathsf{ITEM}(?\mathsf{J}) \wedge \mathsf{hasState}(?\mathsf{J},\top) \wedge \mathsf{hasTime}(?\mathsf{J},\ ?\mathsf{t}_J) \wedge \\
&(?\mathsf{t} \leftarrow ?\mathsf{t}_I + \delta_6) \wedge (?\mathsf{t} \leqslant ?\mathsf{t}_J) \wedge \\
&\mathsf{ITEM}(?\mathsf{W}) \wedge \mathsf{hasState}(?\mathsf{W},\bot) \wedge \mathsf{hasTime}(?\mathsf{W},\ ?\mathsf{t}_W) \wedge \\
&\mathsf{ITEM}(?\mathsf{Q}) \wedge \mathsf{hasState}(?\mathsf{Q},\top) \wedge \mathsf{hasTime}(?\mathsf{Q},\ ?\mathsf{t}_Q) \wedge \\
&(?\mathsf{f} \leftarrow ?\mathsf{t}_W + \delta_7) \wedge (?\mathsf{f} \leqslant ?\mathsf{t}_Q) \wedge \\
&(?\mathsf{t}_J \leqslant ?\mathsf{t}_Q) \wedge (\mathsf{I} \neq \mathsf{W}) \wedge (\mathsf{J} \neq \mathsf{Q}) \\
&\quad \Longrightarrow (\widetilde{A}_5,\top):\mathsf{hasState} \wedge (\widetilde{A}_5,\mathsf{t}_Q):\mathsf{hasTime}.
\end{aligned} \tag{5.12}$$

`Activity Model` α_6: The sixth activity concerns the preparation of a meal, and it is defined with statement's algebra in $\mathscr{T}_6$ as

$$\begin{aligned}
&\widetilde{A}_6 \vDash \left(D_8^\top \vee D_9^\top \vee D_{10}^\top\right) \leqslant \left(\widetilde{C} \wedge \widetilde{C}\right) \leqslant \left(D_8^\top \vee D_9^\top \vee D_{10}^\top\right), \\
&\text{where } \widetilde{C} \vDash \left(\left(I_1^\bot \vee I_2^\bot \vee I_7^\bot\right) + \delta_8\right) \leqslant \left(I_1^\top \vee I_2^\top \vee I_7^\top\right).
\end{aligned} \tag{5.13}$$

$\widetilde{A}_6$ can be expressed as

"doors were opened, two objects were taken,
then after δ_8 time units the objects were released,
then the doors got closed,"

where the two *objects* are associated to prior knowledge concerning sensors related to COOKING and their locations, *i.e.*, (5.4) and Table 5.1. The implementation of (5.13) based on SWRL rules is the following

$$\begin{aligned}
&\mathsf{DOOR}(?\mathsf{O}) \wedge \mathsf{hasState}(?\mathsf{O},\top) \wedge \mathsf{hasTime}(?\mathsf{O},\ ?\mathsf{t}_O) \wedge \\
&\mathsf{DOOR}(?\mathsf{C}) \wedge \mathsf{hasState}(?\mathsf{C},\bot) \wedge \mathsf{hasTime}(?\mathsf{C},\ ?\mathsf{t}_C) \wedge \\
&(?\mathsf{t}_O \leqslant ?\mathsf{t}_C) \wedge
\end{aligned}$$

$$
\begin{aligned}
&\mathsf{ITEM}(?\mathsf{I}) \wedge \mathsf{hasState}(?\mathsf{I},\bot) \wedge \mathsf{hasTime}(?\mathsf{I},\ ?\mathsf{t}_I) \wedge \\
&\mathsf{ITEM}(?\mathsf{J}) \wedge \mathsf{hasState}(?\mathsf{J},\top) \wedge \mathsf{hasTime}(?\mathsf{J},\ ?\mathsf{t}_J) \wedge \\
&(?\mathsf{t} \leftarrow ?\mathsf{t}_I + \delta_8) \wedge (?\mathsf{t} \leqslant ?\mathsf{t}_J) \wedge \\
&\mathsf{ITEM}(?\mathsf{W}) \wedge \mathsf{hasState}(?\mathsf{W},\bot) \wedge \mathsf{hasTime}(?\mathsf{W},\ ?\mathsf{t}_W) \wedge \\
&\mathsf{ITEM}(?\mathsf{Q}) \wedge \mathsf{hasState}(?\mathsf{Q},\top) \wedge \mathsf{hasTime}(?\mathsf{Q},\ ?\mathsf{t}_Q) \wedge \\
&(?\mathsf{f} \leftarrow ?\mathsf{t}_W + \delta_7) \wedge (?\mathsf{f} \leqslant ?\mathsf{t}_Q) \wedge \\
&(\mathsf{I} \neq \mathsf{W}) \wedge (\mathsf{J} \neq \mathsf{Q}) \ \wedge (?\mathsf{t}_O \leqslant ?\mathsf{t}_I) \wedge (?\mathsf{t}_O \leqslant ?\mathsf{t}_W) \wedge \\
&(?\mathsf{t}_J \leqslant ?\mathsf{t}_C) \wedge (?\mathsf{t}_Q \leqslant ?\mathsf{t}_C) \\
&\quad \Longrightarrow (\widetilde{A}_6,\top):\mathsf{hasState} \wedge (\widetilde{A}_6,\mathsf{t}_R):\mathsf{hasTime}.
\end{aligned} \tag{5.14}
$$

`Activity Model` α_7: The seventh activity is cleaning the apartment, which is defined using statement's algebra in $\mathscr{T}_7$, as

$$
\begin{aligned}
&\widetilde{A}_7 \vDash D_{11}^{\top} \leqslant \left(\widetilde{H} \wedge \widetilde{Q}\right) \leqslant D_{11}^{\bot}, \ \text{where} \\
&\quad \widetilde{H} \vDash M_{\{6:10\}} \circ_{h_9} \delta_9, \\
&\quad \widetilde{H} \vDash M_{\{16:18\}} \circ_{h_{10}} \delta_{10},
\end{aligned} \tag{5.15}
$$

$\widetilde{A}_7$ can be expressed as

"the door was opened, then the person stayed δ_9 time units in the leaving room and δ_{10} time units in the kitchen,
then the door got closed."

Here, the door is associated with prior knowledge concerning a CABINET, while $M_{\{6:10\}}$ are related to the LIVINGROOM, and $M_{\{16:18\}}$ to the KITCHEN. All those sensors are represented by the class CLEANING in $\mathscr{L}$, in accordance with (5.4) and Table 5.1. The SWRL-based implementation of (5.15) applied to $\mathscr{T}_7$ is

$$
\begin{aligned}
&\mathsf{DOOR}(?\mathsf{O}) \wedge \mathsf{hasState}(?\mathsf{O},\top) \wedge \mathsf{hasTime}(?\mathsf{O},\ ?\mathsf{t}_O) \wedge \\
&\mathsf{DOOR}(?\mathsf{C}) \wedge \mathsf{hasState}(?\mathsf{C},\bot) \wedge \mathsf{hasTime}(?\mathsf{C},\ ?\mathsf{t}_C) \wedge \\
&(?\mathsf{t}_O \leqslant ?\mathsf{t}_C) \wedge \\
&\mathsf{CLEANED}(?\widetilde{H}) \wedge \mathsf{hasState}(?\widetilde{H},\top) \wedge \mathsf{hasTime}(?\widetilde{H},\ ?\mathsf{t}_{\widetilde{H}}) \wedge \\
&\mathsf{CLEANED}(?\widetilde{Q}) \wedge \mathsf{hasState}(?\widetilde{Q},\top) \wedge \mathsf{hasTime}(?\widetilde{Q},\ ?\mathsf{t}_{\widetilde{Q}}) \wedge \\
&(?\mathsf{t}_O \leqslant ?\mathsf{t}_{\widetilde{H}}) \wedge (?\mathsf{t}_{\widetilde{H}} \leqslant ?\mathsf{t}_C) \wedge (?\mathsf{t}_O \leqslant ?\mathsf{t}_{\widetilde{Q}}) \wedge (?\mathsf{t}_{\widetilde{Q}} \leqslant ?\mathsf{t}_C)
\end{aligned}
$$

$$\Longrightarrow (\widetilde{A}_7,\top):\mathsf{hasState} \wedge (\widetilde{A}_7,\mathsf{t}_C):\mathsf{hasTime}. \tag{5.16}$$

Similarly to the model $\widetilde{A}_3$, we compute the convolution through a computation performed by $\mathcal{M}_7$ before applying the rule. Such operations are used to generate statements $\widetilde{H}$ and $\widetilde{O}$ in $\mathscr{T}_7$ as instances of the class CLEANED, which represent that a location has been cleaned.

`Activity Model` α_8: The eighth activity is choosing an outfit, and it involves $\mathscr{T}_8$. The implementation based on statement's algebra of the related model is

$$\begin{aligned}\widetilde{A}_8 \vDash \left(D_{12}^{\top} + \delta_{11}\right) &\leqslant \left(M_{21}^{\top} \vee M_{22}^{\top} \vee M_{23}^{\top}\right) \\ &\leqslant \left(M_3^{\top} \vee M_4^{\top} \vee \ldots \vee M_9^{\top}\right).\end{aligned} \tag{5.17}$$

$\widetilde{A}_8$ can be expressed as

"a door was opened, then after δ_{11} time units, the person was in $M_{\{21:23\}}$ and then in $M_{\{3:9\}}$,"

where the door is associated with prior knowledge concerning the CABINET2 (shown in Figures 5.6 and 5.7), and we consider

$$\begin{aligned}&\{\mathsf{M}_{21},\ldots,\mathsf{M}_{23}\}:\mathsf{CHOOSE} \sqsubseteq (\mathsf{OUTFIT} \sqcap \mathsf{MOTION}),\\ &\{\mathsf{M}_3,\ldots,\mathsf{M}_9\}:\mathsf{LEAVE} \sqsubseteq (\mathsf{OUTFIT} \sqcap \mathsf{MOTION}).\end{aligned} \tag{5.18}$$

All those sensors concern the class OUTFIT, which is defined in $\mathscr{L}$ in accordance with (5.4) and Table 5.1. Similarly to activity α_3, we specify sub-areas in the corridor and in the living room, *i.e.*, where the person should CHOOSE the outfit and LEAVE it. The latter classes are not further specified as *chosen* and *left* since $\widetilde{A}_8$ do not involve convolution, *i.e.*, no extra computation apart from OWL reasoning is required to compute. The implementation of (5.17) based on SWRL is the rule

$$\begin{aligned}&\mathsf{DOOR}(?\mathsf{O}) \wedge \mathsf{hasState}(?\mathsf{O},\bot) \wedge \mathsf{hasTime}(?\mathsf{O},\ ?\mathsf{t}_O) \wedge\\ &\mathsf{CHOOSE}(?\mathsf{U}) \wedge \mathsf{hasState}(?\mathsf{U},\top) \wedge \mathsf{hasTime}(?\mathsf{O},\ ?\mathsf{t}_U) \wedge\\ &\mathsf{LEAVE}(?\mathsf{V}) \wedge \mathsf{hasState}(?\mathsf{V},\top) \wedge \mathsf{hasTime}(?\mathsf{O},\ ?\mathsf{t}_V) \wedge\\ &(?\mathsf{t}_O \leqslant ?\mathsf{t}_U) \wedge (?\mathsf{t}_O \leqslant ?\mathsf{t}_V) \wedge (?\mathsf{t}_U \leqslant ?\mathsf{t}_V)\\ &\quad\Longrightarrow (\widetilde{A}_8,\top):\mathsf{hasState} \wedge (\widetilde{A}_8,\mathsf{t}_V):\mathsf{hasTime}.\end{aligned} \tag{5.19}$$

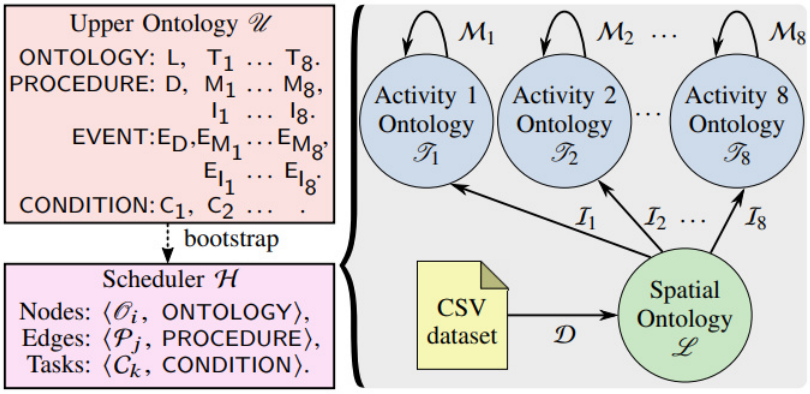


Figure 5.8 On the right hand side is the ontology network for the HAR system presented in Section 5.2 and developed based on Arianna+ for recognizing activities in the CASAS dataset. This network is bootstrapped from the upper ontology shown on the left hand side.

Experiment

To model the scenario outlined above, we configure the network as shown in Figure 5.8 through a specific configuration of the upper ontology $\mathscr{U}$. In particular, we configure $\mathscr{U}$ with an instance L:ONTOLOGY representing an ontology that describes a topological map as shown in Figure 5.6. We configure $\mathscr{U}$ with instances T_a:ONTOLOGY, with $a = 1,\ldots,8$, each representing an ontology describing statements over time, *i.e.*, based on Allen's Algebra, specifically for each activity $\alpha_1,\ldots,\alpha_8$. Consequently, we define in $\mathscr{U}$ eight M_a:PROCEDURE instances, which implement computational procedures evaluating a fluent model for each a-th activity. Each M_a is coupled with another instance I_a:PROCEDURE, which imports statements from L to T_a when particular events occur. In this scenario, we also define in $\mathscr{U}$ an instance D:PROCEDURE, which reads the datasets and creates statements in the spatial ontology L, thus simulating an event-based data stream as it would be generated by sensors.

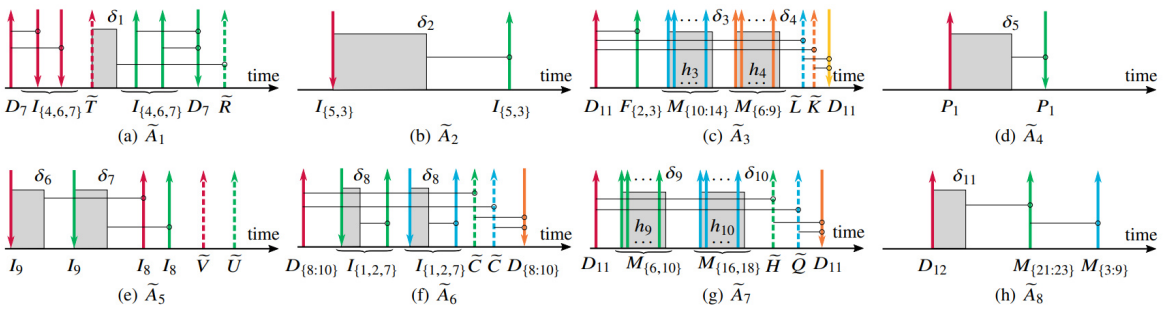


Figure 5.9 A graphical representation of the fluent models to classify the activities in the CASAS dataset.

During bootstrapping, the scheduler generates a network made with the ontologies $\mathscr{L}$ and $\mathscr{T}_a$, respectively specified by the instances L and T_a in $\mathscr{U}$. Therefore, the set of ontologies is $\omega = \{\mathscr{U}, \mathscr{L}, \mathscr{T}_1, \mathscr{T}_2, \ldots, \mathscr{T}_8\}$, whereas the set of procedures is $\rho = \{\mathcal{H}, \mathcal{D}, \mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_8, \mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_8\}$, which includes the scheduler $\mathcal{H}$ itself, and the procedures associated with the instances D, I_a and M_a respectively. The procedures also specify events, which are associated with C_k conditions (3.14), (3.15), (3.16).

The following paragraphs introduce the spatial ontology $\mathscr{L}$, the computational procedures $\mathcal{M}_a$ (which implements fluent models), the activity ontologies $\mathscr{T}_a$, and the scheduling events.
Working of the spatial ontology: $\mathscr{L}$ contextualises spatial information, and it contains only the most up to date statements generated by each sensor in the smart home since it overwrites previous values for those statements. Therefore, the number of axioms does not grow over time. This is a crucial feature since it allows to synchronise the procedures in the network, as it is possible to identify *a priori* a reasoning time limit for the spatial classification of statements. In $\mathscr{L}$ are defined such spatial roles as isIn and $\mathsf{isNearTo}$, which stand for sensory $\mathsf{STATEMENTS}$ related to concepts referring to *locations* (*e.g.*, $\mathsf{KITCHEN}$) and *furniture* (*e.g.*, TABLE). When a sensor generates new data, $\mathcal{D}$ creates a new statement in $\mathscr{L}$, and it updates the corresponding OWL reasoner, which thus infers the person's location and the furniture that he (she) might be using. As we will see in later paragraphs, each computational procedure $\mathcal{I}_a$ is scheduled based on the locations and furniture that we define to be relevant for the a-th activity.
Working of the fluent model: fluent models are based on the formalism presented in Section 3.3.2, which is also graphically shown in Figure 5.9, where statements are depicted as vertical arrows pointing upwards to indicate a $\top$ state and downwards to indicate $\bot$ state. The Figure shows the statements with respect to a relative and qualitative temporal axis, which defines the restriction making a fluent model generate an aggregated statement $\boldsymbol{s}(\widetilde{A}_a) = \top$, thus indicating that the a-th activity has been performed at time $\boldsymbol{t}(\widetilde{A}_a)$. Solid arrows represent raw sensor statements, whereas dashed arrows indicate aggregated statements, *i.e.*, the results of the aggregation of other statements from a procedure. Colours indicate sub-models, *i.e.*, statements in the sub-models respect the precedence operators in $\boldsymbol{f}$. Figure 5.9 shows time intervals of a specified span δ with grey boxes and denotes with h the minimum number of statements expected to be in an interval. Through horizontal lines, the Figure depicts restrictions over time among the statements that must hold for satisfying a model, where the cycled statement is required to be always after the other.

As an example, Figure 5.9b represents the model

$$\widetilde{A}_2 \vDash \left(I^{\perp}_{\{5,3\}} + \delta_2\right) \leqslant I^{\top}_{\{5,3\}}, \tag{5.20}$$

which is satisfied when the item observed by sensors I_5 or I_3 has a $\perp$ state and, after a δ_2 span of time, it becomes $\top$, which means that the objects needed for watching a DVD have been used *for a while*. Figure 5.9d shows the same model applied to the P_1 sensor for identifying the activity of conversing using the phone, and Figure 5.9h shows a similar model for outfit selection. In this case, we assume that an outfit is chosen when the wardrobe door is open, and after a δ_{11} time interval, the person is still in the corridor, *i.e.*, the region associated with sensors M_{21}, M_{22} and M_{23}. We assume that the person places the selected outfit on the sofa, therefore we specify that in order to perform the activity the person should visit the region monitored by M_3 up to M_9. In Section 3.3.2, as examples, we discussed the models $\widetilde{A}_1$ and $\widetilde{A}_7$, which are related to medical dispenser filling and the cleaning activity, respectively. Similarly to the case of $\widetilde{A}_1$, Figure 5.9e represents $\widetilde{A}_5$, which is related to the activity of writing a card. $\widetilde{A}_5$ involves using two objects related to sensors I_8 or I_9 from some time, *i.e.*,

$$\widetilde{A}_5 \vDash \widetilde{U} \wedge \widetilde{V} \vDash \left(\left(I^{\perp}_8 + \delta_6\right) \leqslant I^{\top}_8\right) \wedge \left(\left(I^{\perp}_9 + \delta_7\right) \leqslant I^{\top}_9\right). \tag{5.21}$$

Furthermore, $\widetilde{A}_3$ is recognised if the person remains in some particular location based on the convolution operator, *i.e.*,

$$\begin{gathered} \widetilde{A}_3 \vDash D^{\top}_{11} \leqslant F^{\top}_{\{2,3\}} \leqslant \left(\widetilde{G} \wedge \widetilde{E}\right) \leqslant D^{\perp}_{11}, \\ \text{where} \quad \widetilde{G} \vDash M_{\{6:9\}} \circ_{h_3} \delta_3 \quad \text{and} \quad \widetilde{E} \vDash M_{\{10:14\}} \circ_{h_4} \delta_4. \end{gathered} \tag{5.22}$$

$\widetilde{A}_3$ represents the activity of watering the plants, which can be graphically shown to domain experts as in Figure 5.9c or be expressed in natural language as

"D_{11} was opened and $F_{\{2,3\}}$ was used, then the person stayed in $M_{\{6:9\}}$ for δ_3 time units, and in $M_{\{10:14\}}$ for δ_4 time units, then D_{11} got closed,"

where D_{11}, $F_{\{2,3\}}$, $M_{\{6:9\}}$ and $M_{\{10:14\}}$ are statements in $\mathscr{L}$ and $\mathscr{T}_3$, which are based on *a priori* knowledge about the sensors. In $\mathscr{L}$, they are contextualised in terms of locations and furniture, *i.e.*, respectively CABINET, SINK, PLANT1 and PLANT2, which are involved while watering the plants. In $\mathscr{T}_3$, sensor statements are contextualised with the prior knowledge required only to evaluate $\widetilde{A}_3$ (5.22), *i.e.*, sensor types as {DOOR,FLOW,WATERED} and time intervals δ_3 and δ_4.

For all the eight activities, this section highlights intelligibility of the prior knowledge and of the statement's algebra, as it is presented above for $\widetilde{A}_3$. It is noteworthy that Arianna+ contextualises statements based on a general-purpose hierarchy of DL concepts, and it is possible to further specify their semantics within a context, *e.g.*, that "a person is near PLANT1", or that "a plant has been WATERED", based on locations or time, respectively. As long as an ontology remains small, a detailed classification of statements is intelligible and simple; also, the reasoning complexity is limited.

Working of events: they are bootstrapped from the upper ontology into a set of tasks that periodically evaluate the outcome of each condition C_k. The upper ontology describes the $\mathcal{D}$ procedure using an instance represented with a role $(\mathsf{D},\mathsf{E}_\mathsf{D})$:requires, where E_D is an event occurring once after the bootstrap phase.

The events activating a procedure $\mathcal{I}_a$ depend on the knowledge required by the a-th activity model. For the $\widetilde{A}_1$ model, $\mathcal{I}_1$ maps statements from $\mathscr{L}$ to $\mathscr{T}_1$ when the person is in the kitchen. For this to happen, the event $\mathsf{E}_{\mathsf{I}_1}$ is triggered based on the spatial ontology $\mathscr{L}$. In other words, $\mathcal{I}_1$ depends only on an event composed of a C_1:CONDITION, which has a $\top$ outcome if a number of sensors placed in the living room have a $\top$ state. When $\mathcal{I}_1$ is activated, it retrieves statements D_7, I_4, I_6, I_7 from $\mathscr{L}$, and maps them into $\mathscr{T}_1$, whose purpose is to contextualise knowledge over time to eventually generate a $\top$ outcome for the aggregated statement $\widetilde{A}_1$ (Figure 5.9a). Remarkably, we used the same modular approach for all the eight activities in the dataset, where the set of statements retrieved from $\mathscr{L}$ and mapped into $\mathscr{T}_a$ change with respect to the knowledge strictly required to evaluate the a-th activity model (Figure 5.9).

We rely on similar events for all the $\mathcal{I}_a$ procedures but, since each activity model requires different contextualised statements, their spatial semantics vary. In particular, the scheduler activates the procedure $\mathcal{I}_a$ based on a $\mathsf{E}_{\mathsf{I}_a}$:EVENT such that there is a $\top$ outcome when the person is $(\mathsf{E}_{\mathsf{I}_2})$ in the LIVINGROOM, $(\mathsf{E}_{\mathsf{I}_3})$ near to $\mathsf{CABINET}_1$, or to the SINK, or in the LIVINGROOM, $(\mathsf{E}_{\mathsf{I}_4})$ near to TABLE_2, $(\mathsf{E}_{\mathsf{I}_5})$ near to TABLE_1, $(\mathsf{E}_{\mathsf{I}_6})$ in the KITCHEN, $(\mathsf{E}_{\mathsf{I}_7})$ in the LIVINGROOM, or in the KITCHEN, and $(\mathsf{E}_{\mathsf{I}_8})$ in the CORRIDOR, or near the SOFA or TABLE_1, where the person should leave the selected outfit (Figure 5.6).

The procedure $\mathcal{M}_a$ is based on synthetic events that $\mathcal{I}_a$ generates after importing new statements from $\mathscr{L}$ to $\mathscr{T}_a$. In particular, $\mathcal{I}_a$ generates that event by storing in $\mathcal{T}_a$ a specific $\top$ statement[4], which is reset to $\bot$ by $\mathcal{M}_a$ at the beginning of its computation. This and other similar approaches can always be used to synchronise two procedures in a network.

[4]The statement is expressed as N in Table 5.1.

Result

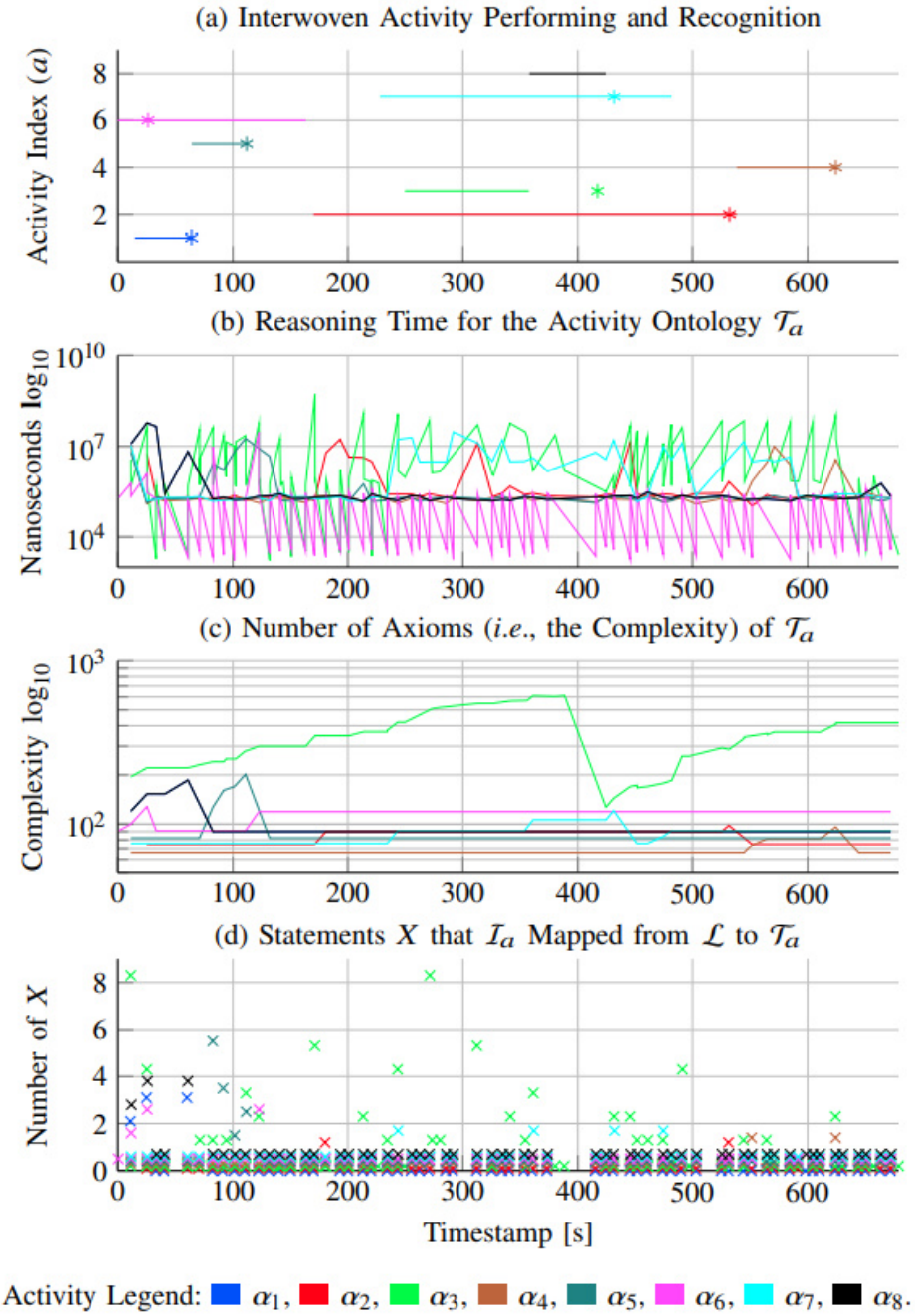


Figure 5.10 The following figures show behaviour of the system. Graph (a) shows the experiment and the associated activity recognitions. Graphs (b) and (c) show the reasoning performance. Graph (d) shows the events in the network and the statements X propagated from the spatial ontology $\mathscr{L}$ to the relevant activity ontology $\mathscr{T}_a$.

Computational performance: Figure 5.10 shows the behaviour of the network of ontologies presented previously. They are running on an Intel i5 2.53 GHz powered workstation with 4 Gb of memory. The Figure shows four graphs with the same timestamp on the x axis and a colour map whereby each colour indicates an activity. The y axis of the first plot represents the activity labels given in the dataset with a horizontal line, *i.e.*, the time spent in performing an activity, and the $*$ symbol indicates the time instant when Arianna+ notifies that the activity has been executed, *i.e.*, $\boldsymbol{t}(\widetilde{A}_a)$ when $\boldsymbol{s}(\widetilde{A}_a) = \top$. The second and the third sub-plots present the reasoning performance for each activity ontology $\mathscr{T}_a$ on a logarithmic scale. Specifically, the plots show the computation time in nanoseconds, and the ontology complexity (*i.e.*, the number of axioms in the ontology), respectively. Figure 5.10d shows the number of statements propagated from the spatial ontology $\mathscr{L}$ to $\mathscr{T}_a$ via the importing procedure $\mathcal{I}_a$.

From a comparison of the four sub-plots, we can analyse the reasoning behaviour with respect to the statements contextualised in the spatial ontology $\mathscr{L}$. The scheduler evaluates statements to detect events and trigger a procedure $\mathcal{I}_a$, which in turn retrieves other statements from $\mathscr{L}$ and adds them to $\mathscr{T}_a$. The number of mapped statements are shown with a $\times$ point in Figure 5.10d, while the complexity of $\mathscr{T}_a$ is shown in Figure 5.10c. Figure 5.10b shows the computation time required to process the statements. Figure 5.10a shows when the models are satisfied with respect to the activity's ground-truth, from which we can also notice the interruptions occurring in the activities.

Figure 5.10b shows a frequent pattern as the conditions are evaluated at a rate of 50 Hz, which drives the event evaluation and, consequently, the scheduling of all the $\mathcal{I}_a$ and $\mathcal{M}_a$ procedures. Depending on the context over time, not all events generate statements that get evaluated with the procedure $\mathcal{M}_a$, and this leads to the spikes shown in Figure 5.10b. In particular, the scheduler synchronously activates $\mathcal{M}_a$ when a statement is mapped by $\mathcal{I}_a$. As a consequence, $\mathcal{M}_a$ calls the reasoner associated with the $\mathscr{T}_a$ ontology, which evaluates activity models α_a based on a temporal representation.

Since we used stream reasoning techniques [141, 142], the reasoning complexity depends on an ontology's previous state, and the time required to check whether statements in $\mathscr{T}_a$ satisfy an activity model is dependent on the context over time. For this reason, the characterisation of the reasoning complexity for an ontology is far from trivial, and we use the number of axioms that it contains as an approximation. However, this issue is limited to the spatial ontology $\mathscr{L}$, whose reasoner (invoked by the $\mathcal{D}$ procedure) evaluates all the sensor statements over time. Figure 5.11 shows that the reasoning complexity of the spatial ontology is similar to the one characterising activity ontologies, although with a smaller

variance. Also, Figure 5.11 shows that the number of axioms used to represent prior spatial knowledge without any sensor statements is 273, *i.e.*, the lower complexity bound of $\mathscr{L}$. Instead, the upper bound of 353 axioms shows the maximum complexity of $\mathscr{L}$, and it cannot increase due to the overwriting policy of $\mathscr{D}$.

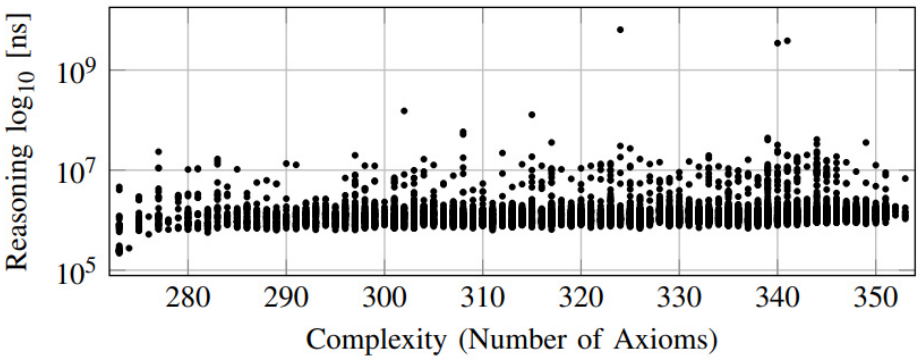


Figure 5.11 The spatial ontology $\mathscr{L}$ reasoning time against ontology complexity.

In Figures 5.10c, we observe an exponential increase of the complexity over time for not well contextualised activities. For instance, α_3 relies on a model that requires statements from many locations in the apartment, *i.e.*, the spatial context related to the model $\widetilde{A}_3$ (Figure 5.9) is not *well-separated* enough from other activities as far as its definition is concerned. Instead, for activities characterised by a more distinct context, *e.g.*, α_2 and α_4, the complexity remains limited. In this experiment, we delete all the statements from $\mathscr{T}_a$ when the a-th activity has been recognised. The effect of this can be seen by noticing the drop of the curves in Figure 5.10c with respect to the instants when an activity is recognised, as shown in Figure 5.10a. This statements' cleaning procedure is associated with tackling the issue of continuous reasoning that often leads to an exponential increase in the complexity [143, 144].

Due to an increase in the amount of knowledge to process, reasoning becomes exponentially complex, and this can also be noticed by comparing the activities α_3 and α_6 in Figure 5.10. In this case, α_6 requires less computation time than α_3, since the former has a well-defined spatial contextualisation, while the latter occurs in a not well determined area. Therefore, having a small and well contextualised ontology $\mathscr{T}_a$ decreases the computational load, and it is more likely that the related model is processed only when required. We tested the performance of the network by accelerating the simulation 4× faster than its original speed, and we did not observe notable changes in the activity recognition rate. Instead, for higher speeds, the recognition performance drastically decrease due to the overwriting policy of $\mathscr{D}$.

	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8
α_1	0.95	0	0	0	0	0	0	0
α_2	0	0.95	0.05	0	0	0	0.05	0
α_3	0	0	0.7	0	0	0	0.05	0
α_4	0	0	0	1	0	0	0.05	0
α_5	0	0	0	0	0.8	0	0	0
α_6	0	0	0	0	0	0.7	0.05	0
α_7	0	0	0.25	0	0	0	0.8	0
α_8	0	0	0	0	0	0	0	0.95
unclassified	0.05	0.05	0	0	0.2	0.3	0	0.05

Table 5.2 Confusion matrix of the activity recognition rate obtained with the ontology network.

We attempted to test an ontology network made of a single node such that it encodes time intervals, spatial knowledge, and all the activity models. Our objective was to compare the reduction of computational load between an approach that uses a single ontology and the ontology network we develop. Unfortunately, we cannot report the comparison because we experienced *out of memory* issues with our machine. In the HAR system evaluated in Section 5.1, we used Arianna+ to develop another ontology network designed for a scenario similar to the one presented in this work but involving less activities and sensors. In that evaluation, we observed that the reasoning time scales exponentially when a single ontology becomes more complex, whereas the overall reasoning time scaled linearly with that network. Although, that evaluation requires more extensive testing to fully characterise the complexity of a general ontology network designed with Arianna+ , with the HAR system's evaluation presented in this section we can confirm a reduction of the computational load. Indeed, with a network of small contexts we could recognise activities with *soft* real-time constrains, whereas if the same knowledge is encoded in a single ontology, we could not process the data stream.

Activity recognition accuracy: Table 5.2 shows the activity recognition confusion matrix. Table 5.3 whose last column is deduced from Table 5.2 shows the F-measure of the activity recognition rates compared among ours and other approaches for the same dataset. The other techniques are based on ML and probabilistic approaches, as well as on context-based systems exploiting hybrid OWL-statistical reasoning. As mentioned above, Table 5.3 does not indicate that the activity models we developed in Section 5.2.3 are expected to outperform other approaches in a less controlled scenario. Our aim is to ground the possibility that having multiple ontologies does not jeopardise, in principle, the possibility of obtaining results comparable to other methods. These techniques approach the problem with a different paradigm, *i.e.*, they are inclined towards designing a *single context* to ground all the activity

	[145]	[140]	[146]	[147]	[148]	[139]	[94]	[97]	Arianna+
α_1	.66	.66	.77	.77	.85	.95	.68	.89	.97
α_2	.69	.86	.73	.78	.81	.97	.98	.91	.92
α_3	.68	.29	.75	.79	.72	.94	.30	.72	.80
α_4	.66	.59	.75	.78	.72	.95	.87	.34	.98
α_5	.61	.83	.72	.79	.81	.98	.98	.78	.89
α_6	.65	.83	.75	.79	.88	.97	.74	.98	.80
α_7	.64	.88	.72	.78	.57	.96	.76	.84	.78
α_8	.66	.67	.72	.79	.88	.95	1.0	1.0	.97

[145] Artificial Neural Network, [140] Hidden Markov Model,
[146] Support Vector Machine, [147] Bayesian network,
[148] MLN with numerical constraints, [139] MLN extended with OWL,
[94] Hybrid OWL-statistical system, [97] OWL Class Expression Learning.

Table 5.3 F-measure of activity recognitions shown for the ontology network developed with Arianna+ and approaches using a single context to represent data.

models in the dataset. Instead, our paradigm involves hierarchical data representations aimed to accommodate each activity model within a purposely-tailored context.

Table 5.3 shows that, in this scenario, symbolic reasoning based on hierarchical and concurrent contexts can exhibit results comparable with other state of the art approaches. However, our activity models require an engineering process that is hard to generalise, especially due to the limited size of the dataset. Hence, we cannot use the comparison to evaluate the recognition rate of specific activities. Nevertheless, considering that Arianna+ has been designed to support an iterative development process, and that the network presented in this section has been implemented with simple heuristics, Table 5.3 confirms that managing hierarchical and concurrent contextualisations of data could be effective for human activity recognition. This is true if suitable *a priori* knowledge is encoded in Arianna+ , which highlights the need for undertaking a development process guided by domain experts.

5.2.4 Summary

This work evaluates a HAR system based on Arianna+ using the CASAS dataset. The motivation for this evaluation is to assess the modularity, intelligibility and robustness of a system based on Arianna+. Hence a system is developed with an ontology network wherein there are nine core ontologies. One ontology called the `place ontology` is used to spatially relate sensory data and trigger (based on events) temporal reasoning on one or more of the remaining eight activity ontologies. Hence, the reasoning is concurrent and based on a hierarchy of contexts, which are represented as small ontologies. Due to contextualized reasoning computations are performed on minimum relevant data and only when required.

As ontologies can be evaluated concurrently, even concurrent activities can be recognized. Due to these reasons, activity recognition is online. Furthermore, this work highlights intelligibility in the inner-workings of the system by fluent models of all the eight activities and their representation in natural language. This emphasizes the point that modular and intelligible systems can enable ease in an iterative development process in collaboration with health-care domain experts in ADL.

In this implementation, we do not consider the challenging issues related to multi-occupancy, and in this scenario, which uses the CASAS dataset, the system is developed by considering that there is a single person in the smart home. To be able to perform activity recognition with multiple persons, the system would require sensory data tagged with a person's unique identification. If this kind of data is present then the system could do HAR for each new person by instantiating a new network dedicated to each unique person. Nonetheless, the feasibility and computational performance of such an approach is yet to be evaluated. Furthermore, the current implementation of the system does not take into account uncertainties, i.e., it is not robust to missing or noisy information in input data. In order to overcome this limitation, in the future, Arianna+'s design could be further developed for reasoning using fuzzy[81] or probabilistic [82] OWL reasoners. Else instead, a simpler approach could also be applied by having redundant activity models, i.e., multiple activity models that use different techniques to recognize the same activity. For instance, ML could effectively be used to generate discrete events by processing a sensory data stream, e.g., the occurrence of actions using wrist-related inertial data [18]. Based on a predefined heuristic the most accurate result (between multiple activity models recognizing the same activity) can be considered as the true one. Nonetheless, dealing with missing or noisy sensor data would require incorporation of fuzzy or probabilistic reasoners.

Moreover, this work also does not deal with the issue of variability in the execution of sub-activities or actions of a composite ADL. In this work, the system's accuracy of activity recognition - for half of the activities in the CASAS dataset (i.e., $\alpha_1, \alpha_2, \alpha_4, \alpha_8$) - is better than the accuracy of state of the art approaches (see Table 5.3). For the remaining half of the activities, the accuracy of this system is comparable to the accuracy of the state of the art approaches. The reason for the overall good accuracy is that the fluent models are developed precisely by a knowledge engineer to recognize the activities being performed in the dataset. This can be considered as evaluating the system in a controlled environment. Hence, in a less controlled environment, i.e., in the real-world, data-driven approaches may prove to be more accurate than fluent models. Although this is a limitation of the system presented in this Section, it is not a limitation of Arianna+ framework itself as it can accommodate

the outcomes of both knowledge-based and data-driven models in a context-oriented and concurrent way. Finally, although the results of our approach are compared with other state of the art hybrid approaches (see Table 5.3), we restrain from making any conclusive statements about the comparison, because in terms of the approach, all the state of the art approaches are algorithmically hybrid whereas our approach is hybrid in an architectural-sense. Furthermore, the fluent models (i.e., activity models) in our approach are hand-crafted and the activity models in the state of the art approaches are learnt from the dataset.

5.3 A real-world HAR system showcasing explainable results using SPARQL queries

5.3.1 Introduction

This work exploits Arianna+ using OWLOOP API to develop a simple real-world HAR system that saves all activity recognition results in a purposefully designed ontology for querying. This work particularly highlights that given a well-designed knowledge-base, querying is a simple and efficient way to retrieve and explain HAR results. This is particularly relevant to one aspect of the first motivation presented in Section 1.1.1, i.e., to enable enriched collaboration between HAR researchers and health-care domain experts by having an intelligible system. Furthermore, this work explores the research problems highlighted under RQ1, RQ3, and RQ4, which are presented in Section 2.4.

The W3C's initiative of having a semantic web introduced some very well known and established knowledge representation standards, namely, Resource Description Framework (RDF), Resource Description Framework Schema (RDFS) and Ontology Web language (OWL). RDF is a graph-based data model. It represents information as a labeled, directed multi-graph with vertices and labeled edges (multiple edges with different labels between the same nodes are allowed). Vertices consist of Internationalized Resource Identifiers (IRIs) (representing abstract "things"), literals (concrete data values) and blank nodes (dummy "convenience" nodes without explicit IRI). And edges consist of IRIs. An RDF graph is expressed as a set of <subject, predicate, object> triples, each interpreted as an edge labelled with "predicate" going from the "subject" node to the "object" node. RDF data is stored in a native purpose-built database called triplestore. SPARQL is the language dedicated to querying RDF graphs.

RDFS extends RDF with the most basic ontological constraints and semantics: class and property subtypes, along with property range and domain restrictions. These constructs allow

for building very simple type hierarchies over RDF data, which are also represented within RDF graphs. OWL is a family of description-logic-based ontology languages. It adds further ontological constructs on top of those introduced by RDFS. Hence, as OWL is based on RDFS, which is based on RDF, SPARQL can also be used to query knowledge represented using OWL.

As SPARQL is supported by the majority of RDF-compliant frameworks and reasoners, HAR researchers have also attempted to implement systems and frameworks that make use of SPARQL and OWL for activity recognition [79, 149]. The work presented in this chapter, simply uses SPARQL queries to retrieve HAR results in an explainable way. The work also highlights the importance of carefully designing an ontology wherein the knowledge is saved.

5.3.2 Methodology

The methodology of this work has three main components: heterogeneous sensor data observation, development of an ontology network based on Arianna+ and using OWLOOP API, knowledge querying using SPARQL to show intelligibility of HAR results. The HAR system developed in this work is designed to be functional for multiple users. The activities recognized by this system are enumerated as (α_1) having breakfast and (α_2) routine morning hygiene. For simplicity, we consider that having breakfast activity (α_1) occurs when the user is in the kitchen in the morning, is near the kitchen-table, does the pouring action and drinking action at least once, and spends some user-defined amount of time near the kitchen-table. Similarly, we consider that routine morning hygiene (α_1) occurs when the user is in the bathroom in the morning, is near the washbasin, does brushing action, and spends some user-defined time amount of time near the washbasin.

The sensors used in this work are Bluetooth-based Estimote beacons [5] and a smart-watch, for proximity based human localization. Furthermore, accelerometer data streaming from the smart-watch is used for action recognition. Note that a data-driven technique is suitable for training a model or models for recognizing actions such as pouring, drinking, and brushing. In this work we do not train action recognition models as it was out of the main scope of this work. Instead while the volunteer performs the experiment, the system designer simulates the action recognition. While human localization is working based on the volunteers proximity to the Bluetooth beacons.

In this work, new activity ontologies are designed and placing ontology is used from previous work presented in Section 5.1. Furthermore, to show the advantage of using

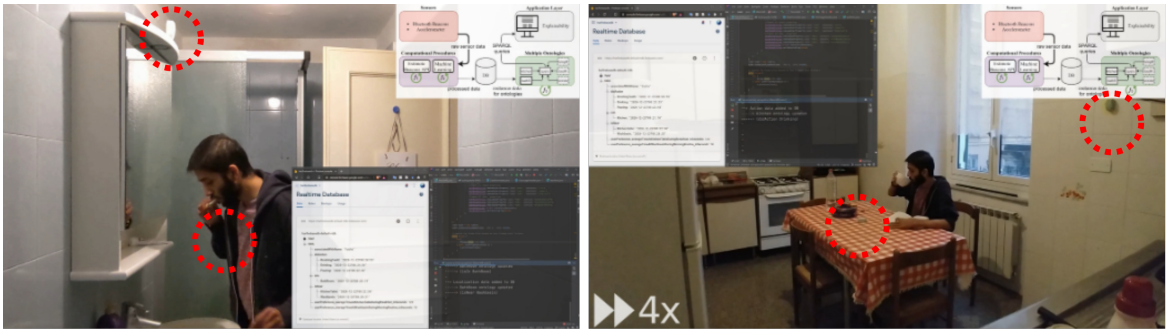
[5] https://estimote.com/

SPARQL for querying HAR results, a special query ontology is designed. The following subsections present the system's architecture, the activity ontologies, the query ontology, the working of the system, and its results.

5.3.3 Experimental evaluation

Experimental setup

The experimental setup is in a small apartment with a living room, bedroom, kitchen and bathroom. This preliminary work showcases activity recognition in the kitchen and in the bathroom. Figure 5.12 shows volunteer performing an activity in the bathroom (see left image) and the kitchen (see right image). In the same figures, Bluetooth beacons and the smartwatch, i.e., the distributed and wearable sensors, are circled in red.



Bathroom Kitchen

Figure 5.12 Volunteer performing an experiment at the smart-home. Left image shows volunteer in the Bathroom. Right image shows volunteer in the Kitchen.

The Figure 5.13 shows the system's architecture. It is composed of five modules. The first module simply depicts the physical sensors, which in our case are Bluetooth beacons and smart-watch. The second module is associated with the computational procedures of the ontology network. They process raw sensor data coming from the first module, generate statements, and save them in the database. The computational procedures have their own independent frequency at which they process raw sensor data to generate sensor events and map them as statements. The raw sensor data is assumed to be post-processed to a point where it is a binary value with its associated label and timestamp. For instance, consider that IMU data from a smart-watch is processed by a computational procedure to a point where a label such as 'pouring' is true or false and has a certain timestamp. This post-processed data is saved in the database as a statement. The third module is the database, which in this system is a real-time database known as Firebase from Google. A key feature of Firebase

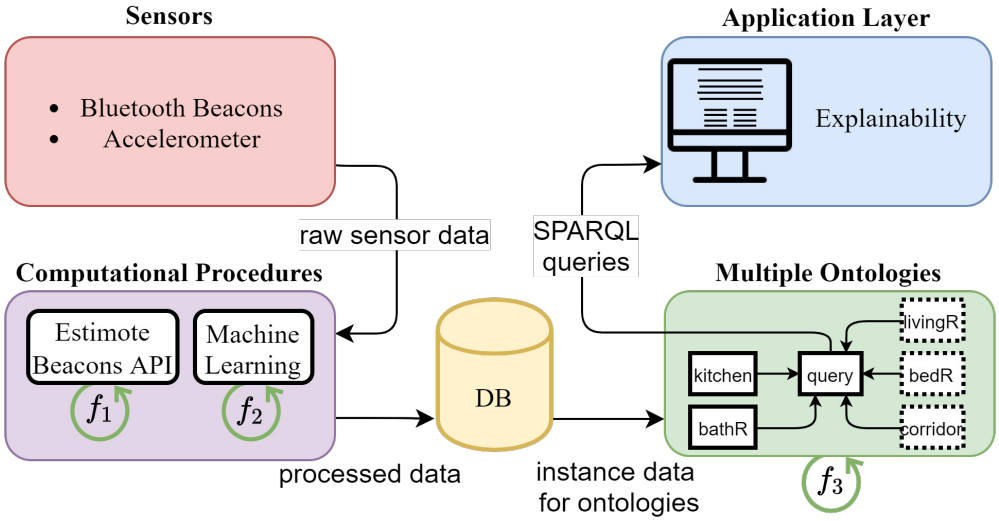


Figure 5.13 The system's architecture for the work presented in Section 5.3.

is its event-listener design patter. By this feature, as soon as a statement gets updated in the Firebase, the relevant ontology in the network consumes the relevant statements for contextualized reasoning. The fourth module is the ontology network, composed of three main ontologies in this system's architecture. They are the kitchen ontology, bathroom ontology, and query ontology. The frequency f_3 associated with this module is 1 hertz, i.e., every second the activity models in the network were getting updated with the current time, and reasoning was performed. If an activity gets recognized by one of the activity models then the result is saved in the query-ontology. The final and fifth module is called the explainability module. This module basically refers to the ability to query the query ontology with SPARQL.

As this work is intended to showcase the intelligibility of HAR results, we particularly develop and highlight three ontologies. Namely, activity ontologies representing α_1 and α_2 activity models, and the query ontology for representing HAR results. Figure 5.14 shows the contents of the kitchen ontology using the Protégé editor. The contents of the bathroom ontology are exactly similar except for the SWRL rule that encodes the fluent model of the activity. Figure 5.15 shows the contents of the query ontology using the Protégé editor. It reuses the T-box of the activity ontology except for one difference, i.e., it has an extra class called `HumanActivityRecognition`, whose instances are the activity recognition resulting statements coming from activity ontologies.

Following are the required statements and fluent models describing the two activities α_1, i.e., having breakfast and α_2, i.e., morning routine hygiene. We also present a simple

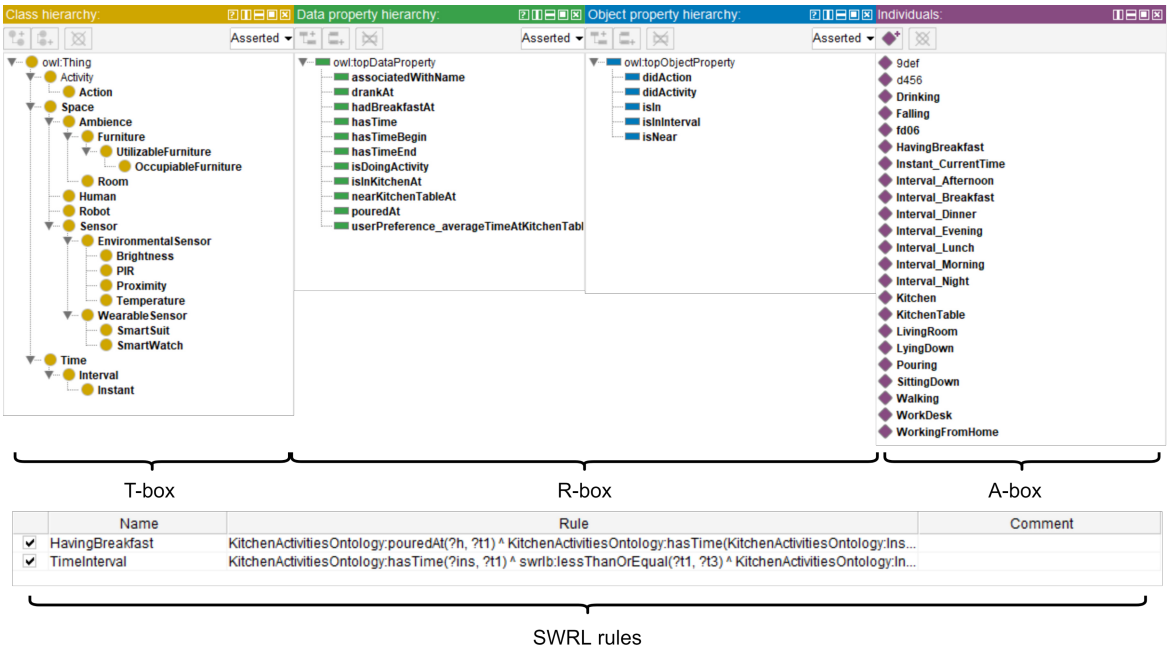


Figure 5.14 Kitchen ontology shown in Protégé editor.

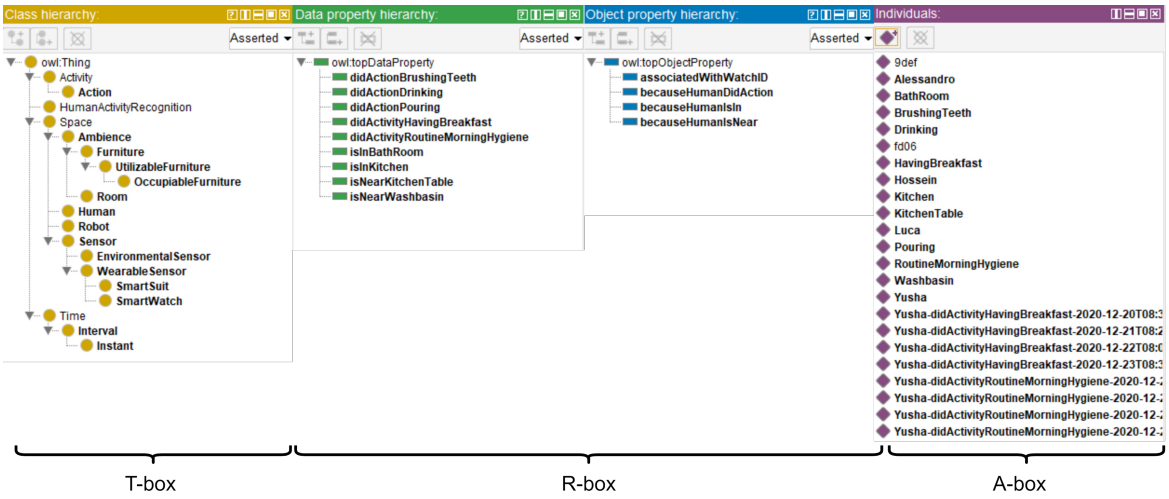


Figure 5.15 Query ontology shown in Protégé editor.

temporal model that is present in both activity ontologies, which infers the correct interval of the current time instant.

`Temporal Model`: Consider that $\widetilde{M}$ represents the statement `(CurrentTimeInstant isInInterval Morning)`, $\widetilde{I}$ represents the statement `(Morning hasBeginTime` $6:00$`)`, $\widetilde{J}$ is the statement that gets updated with the current time, and $\widetilde{K}$ represents the statement `(Morning hasEndTime` $12:00$`)`. According to the statement's algebra, the temporal model is as follows in both the activity models, i.e., α_1 and α_2:

$$\widetilde{M} \vDash I \leqslant J < K \tag{5.23}$$

`Activity Model` α_1: Consider that $\widetilde{A}_1$ represents `(Human(?h) didActivity HavingBreakfast)`, $\widetilde{M}$ represents `(CurrentTimeInstant isInInterval Morning)`, A represents `(Human(?h) isIn Kitchen)`, B represents `(Human(?h) isNear KitchenTable)`, C represents `(Human(?h) didAction Pouring)`, D represents `(Human(?h) didAction Drinking)`, J is the statement that gets updated with the current time, and δ_1 is the statement that holds the user's preference for the time to spend near the `KitchenTable`. According to the statement's algebra, $\widetilde{A}_1$ is as follows:

$$\widetilde{A}_1 \vDash \widetilde{M} \wedge \big(A < B < C < D\big) \wedge \big(B + \delta_1\big) < J \tag{5.24}$$

A visual representation of the statements - that make up the activity model α_1 and the temporal relationships between them - is shown in Figure 5.16.

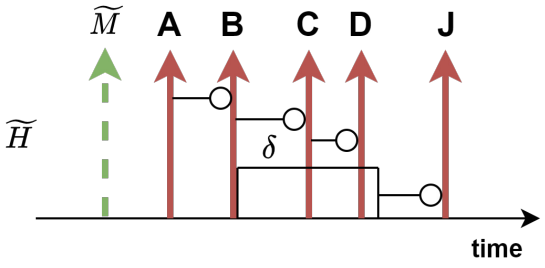


Figure 5.16 Visual representation if the `HavingBreakfast` activity model, i.e., $\widetilde{A}_1$.

`Activity Model` α_2: Consider that $\widetilde{A}_2$ represents `(Human(?h) didActivity MorningRoutineHygiene)`, $\widetilde{M}$ represents `(CurrentTimeInstant isInInterval Morning)`, P represents `(Human(?h) isIn Bathroom)`, Q represents `(Human(?h) isNear Washbasin)`, R represents `(Human(?h) didAction Brushing)`, J is the state-

ment that gets updated with the current time, and δ_2 is the statement that holds the user's preference for the time to spend near the `Washbasin`. According to the statement's algebra, $\widetilde{A}_2$ is as follows:

$$\widetilde{A}_2 \vDash \widetilde{M} \wedge \big(P < Q < R\big) \wedge \big(Q + \delta_2\big) < J \tag{5.25}$$

Experiment

A video was recorded while the volunteer naturally performed the experiment [6]. Link to the video can be found in the footnote. The volunteer performed the activities in the following way, (i) he enters into the kitchen then goes towards the attached bathroom, (ii) he enters into the bathroom and goes neat the washbasin, (iii) he takes the brush and initiates brushing, (iv) while brushing he goes to the kitchen near the sink and takes a cup, (v) goes near the kitchen table and places the cup, (vi) goes back to the bathroom, spends few seconds near the washbasin and finishes brushing the teeth, (vii) he enters the kitchen and starts to prepare hot water, does pouring gesture, (viii) once the hot water is ready he sits at the kitchen table and spends some time here, (ix) does drinking gesture a few times while eating, and the experiment is over.

The morning routine hygiene activity gets recognized after the user spends more than 10 seconds near the washbasin while brushing his teeth, because that is the amount of time the user prefers and has set in the database. The having breakfast activity gets recognized after the user spends more than 2 minutes at the kitchen table while having done the pouring and drinking actions. It is the amount of time the user prefers and has set in the database. In total, the volunteer performed the activities for four days, which can be seen in the result of a query in the next section.

Result

This section presents SPARQL queries, what they mean in natural language, and their proper syntax and results from Protégé. The query language works by doing pattern matching on the RDF triples that exist in the knowledge base. Before writing a query we define some prefixes otherwise the queries would not be easy to naturally comprehend. The prefix that starts with the ':' belongs to the default namespace of the ontology. A basic query is a triple pattern with one or more variables in place of the subject, predicate or object. After the

[6] https://youtu.be/c5TM4__nF8I

`SELECT` keyword we mention the variables and after the `WHERE` keyword we mention the triple patterns (containing the variables we are looking for) to match in the knowledge base.

Note that to query the ontology efficiently, one must be a aware of the terms in the ontology, i.e., the T-box and R-box. The first query is to *"select and list all the activities in the knowledge base"*. Hence, `Query 1` and its result are shown in Figure 5.17. The second query is to *"select and list all the users in the knowledge base"*. Hence, `Query 2` and its result are shown in Figure 5.18. The third query is to find out *"which user did the activity having-breakfast and at what time?"*. Hence, `Query 3` and its result are shown in Figure 5.19. The fourth query is to find out the reason for *"why having-breakfast activity gets recognized?"*. Hence, `Query 4` and its result are shown in Figure 5.20. The fifth query is to *"find out the reasons for a particular HAR result and their time of occurrence*. This query checks the reason and the time at which a particular user performs a particular activity at a particular time. The query also has the keyword `ORDER BY ASC` for the variable `?atDateTime`, this orders the result list with the oldest timestamp as the first element. Hence, `Query 5` and its result are shown in Figure 5.21. Finally, the sixth query is similar to the fifth, except that it is for the morning routine hygiene activity. The `Query 6` and its result are shown in Figure 5.22.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://www.semanticweb.org/Arianna/QueryHumanActivityOntology#>

SELECT ?Activity
WHERE {
        ?Activity rdf:type :Activity.
}
```

Activity
RoutineMorningHygiene
HavingBreakfast

Figure 5.17 SPARQL `Query 1` and its result.

5.3.4 Summary

This work presents a real-world HAR system based on Arianna+ and developed using OWLOOP API. The objective of this work is to showcase a prototype real-world HAR

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://www.semanticweb.org/Arianna/QueryHumanActivityOntology#>

SELECT ?User
WHERE {
        ?User rdf:type :Human.
}
```

User
Alessandro
Luca
Yusha
Hossein

Figure 5.18 SPARQL `Query 2` and its result.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://www.semanticweb.org/Arianna/QueryHumanActivityOntology#>

SELECT ?User ?atDateTime
WHERE {
        ?User :didActivityHavingBreakfast ?atDateTime.
}
ORDER BY DESC(?atDateTime)
```

User	atDateTime
Yusha	"2020-12-23T08:34:07"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Yusha	"2020-12-22T08:05:10"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Yusha	"2020-12-21T08:25:22"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Yusha	"2020-12-20T08:30:05"^^<http://www.w3.org/2001/XMLSchema#dateTime>

Figure 5.19 SPARQL `Query 3` and its result.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://www.semanticweb.org/Arianna/QueryHumanActivityOntology#>

SELECT ?because ?reason
WHERE {
        :HavingBreakfast rdf:type ?Activity.
        ?because rdfs:domain ?Activity.
        :HavingBreakfast ?because ?reason.
}
```

because	reason
becauseHumanIsIn	Kitchen
becauseHumanIsNear	KitchenTable
becauseHumanDidAction	Drinking
becauseHumanDidAction	Pouring

Figure 5.20 SPARQL `Query 4` and its result.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://www.semanticweb.org/Arianna/QueryHumanActivityOntology#>

SELECT ?HAR ?atDateTime
WHERE {
        ?HAR  rdfs:domain :HumanActivityRecognition.
        :Yusha-didActivityHavingBreakfast-2020-12-23T08:34:07 ?HAR ?atDateTime.
}
ORDER BY
ASC(?atDateTime)
```

HAR	atDateTime
isInKitchen	"2020-12-23T08:30:44"^^<http://www.w3.org/2001/XMLSchema#dateTime>
isNearKitchenTable	"2020-12-23T08:32:05"^^<http://www.w3.org/2001/XMLSchema#dateTime>
didActionPouring	"2020-12-23T08:32:23"^^<http://www.w3.org/2001/XMLSchema#dateTime>
didActionDrinking	"2020-12-23T08:33:43"^^<http://www.w3.org/2001/XMLSchema#dateTime>

Figure 5.21 SPARQL `Query 5` and its result.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://www.semanticweb.org/Arianna/QueryHumanActivityOntology#>

SELECT ?HAR ?atDateTime
WHERE {
        ?HAR  rdfs:domain :HumanActivityRecognition.
        :Yusha-didActivityRoutineMorningHygiene-2020-12-23T08:30:09 ?HAR ?atDateTime.
}
ORDER BY
ASC(?atDateTime)
```

HAR	atDateTime
isInBathRoom	"2020-12-23T08:29:53"^^<http://www.w3.org/2001/XMLSchema#dateTime>
isNearWashbasin	"2020-12-23T08:29:55"^^<http://www.w3.org/2001/XMLSchema#dateTime>
didActionBrushingTeeth	"2020-12-23T08:30:06"^^<http://www.w3.org/2001/XMLSchema#dateTime>

Figure 5.22 SPARQL `Query 6` and its result.

system that saves its activity recognition results in a purpose-built query ontology. SPARQL can then be used to query this ontology to retrieve results in an intelligible manner. Explainable results are particularly relevant for an enriched collaboration between HAR system researchers/developers and health-care domain experts.

The HAR system in this work is able to recognize two activities, i.e., having breakfast activity and the morning routine hygiene activity. An ontology network was developed encoding fluent models for these two activities. The experimental setup was made at a volunteers apartment. Bluetooth beacons, namely, the Estimote beacons, were installed as the distributed sensors for human localization based on beacons' proximity to the wrist-worn smart-watch. Developing ML models for recognizing human actions, e.g., pouring, drinking, and brushing, was out of the scope of this work. Hence, a system developer simulated the human action recognition events while watching the volunteer perform the experiment. The volunteer performed the experiment for four days and the results were saved in the query ontology. Finally, this work presented the SPARQL queries, their meaning in natural language, and their intelligible results. In the future, better ways to represent the query ontology can be explored. As the representation of the ontology in which the results are being saved is relevant to the intelligibility of the SPARQL queries and their results. A limitation of this work is that the intelligibility feature of the framework could not be evaluated by discussions with health-care domain experts as their availability was affected by the Covid-19 pandemic [137, 138].

5.4 Online HAR enables a smart-home application with a companion robot

5.4.1 Introduction

This work exploits Arianna+ using OWLOOP API to develop a simple real-world HAR system that can provide online activity recognition knowledge to a companion robot. This work highlights that robust HAR can enable or support many kinds of smart-home applications. Furthermore, this work explores the research problems highlighted under RQ2, RQ3, and RQ4, which are presented in Section 2.4.

In particular for health monitoring and care oriented smart home systems, the type of sensors used and how the data is managed affects the users' perception of privacy and trust [150–152]. Furthermore, the ease of interaction with the system plays an important role in the usability and acceptability of the system [153, 154]. More often than not, health-oriented smart home systems for the elderly do not provide services that are directly-accessible or under-control by the elderly users (for instance, recognized ADL and health insights are intended to be used by care-takers or doctors [8]). Nonetheless, in recent years that has been significant research activity in expanding the ways of interaction between the users and their smart home. In particular, voice interfaces have been rapidly developed [155, 156].

In general AAL smart home systems use data coming from sensors that are installed within the house, i.e., distributed environmental sensors, or sensors worn by the users, i.e., wearable devices. During the development of our smart home system, on the one hand, we avoid sensors that are perceived as 'intruding on privacy' by choosing sensors other than cameras. On the other hand, our smart home system has the ability to proactively interact with the user. This feature is added not just for support and well-being of the user, but also to give the user control and accessibility of the system. It is interesting to point out that "in spite of privacy and security breaches, people frequently compromise their privacy in exchange for certain benefits of a technology or a service" [157]. In this work, we present a simple health-care oriented AAL smart home system developed keeping in mind the above mentioned issues. We use this system to evaluate the user's experience (UX) while s/he interacts with two different interactive devices, i.e., a vocal-assistant or a robot-assistant. The developed system is context-aware and multi-user, i.e., it is able to react according to the context of the user and the reaction can be unique for each user.

Using the developed smart home system we evaluate the main hypothesis of this work, i.e., 'Is there a significant difference in the user experience if the interactive device is a

robot-assistant instead of a vocal-assistant?'. In this work we focus particularly on evaluating the human-robot-interaction (HRI) by surveying the users' experience while they interact with the smart-home when it is equipped with a robot-assistant and when it is equipped with a vocal-assistant. There exist many different surveys for evaluating the UX. Some studies [158–160] have tried to enumerate and describe how surveys can be used for evaluating the UX. Other studies [161–163] have described different ways of designing own and unique UX surveys. For analysing our system, we decided to use the one commonly used by the scientific community, i.e., the user experience questionnaire (UEQ) [164–167]. Using UEQ the UX is evaluated for the case when the smart-home system is equipped with a robot-assistant and for the case when it is equipped with a vocal-assistant.

5.4.2 Methodology

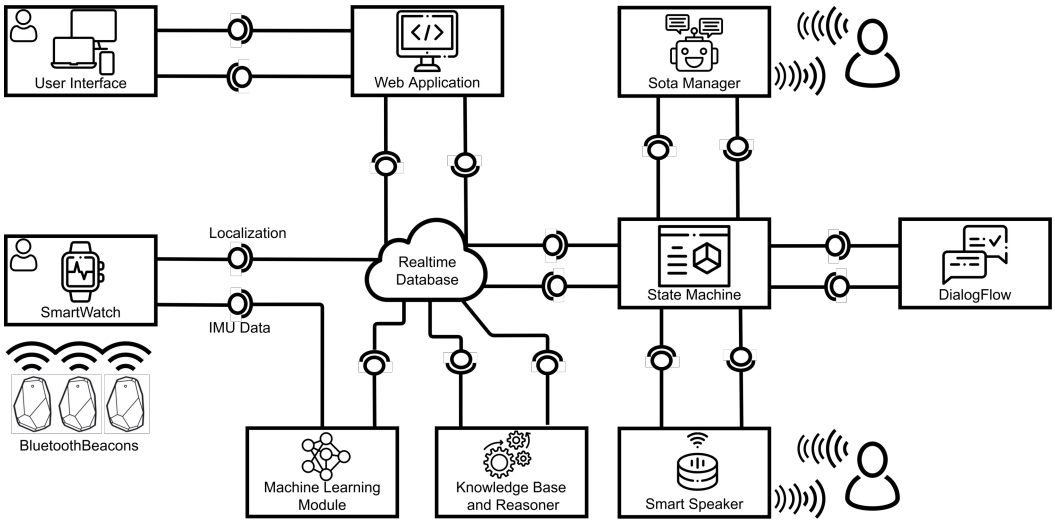


Figure 5.23 The system's architecture for the work presented in Section 5.4.

As shown in Figure 5.23, the system consists of many different and connected parts. Specifically, the system is composed of three macro-layers connected to each other via a database. Firstly, the sensing Layer that deals with the acquisition of sensory data coming from a smart watch. Secondly, the HAR layer deals with modeling and recognizing activities. Lastly, Human Computer Interaction (HCI) Layer that deals with providing interfaces for human computer interaction.

As the system deals with data that is continuously changing, it is necessary that we use a database that has near real-time processing capability. This kind of database is called *real-time database*, and they are commonly used in IoT systems. All the relevant information

are stored in the database, which provides the context regarding where the assisted people are located to any application that requests it. In this way, all our modules are connected with a shared knowledge but most of all it leaves the possibility, for future applications with different purposes to hook on to that knowledge by providing different and new services.

In particular, the smart watch takes care of defining the user's position, based on the proximity to Bluetooth beacons, sending it to the database and sending the raw inertial data to a smart watch which sends information back to a Machine Learning (ML) application for action recognition. The Machine Learning module then receives raw inertial data and detects whether the assisted person is performing the actions for which the neural network has been trained. In this way, the information concerning the users' position and action labels along with the timestamp when they were recognized are sent and saved to the real-time database. This information is then used by the HAR layer, which is an implementation based on Arianna+ described in Chapter 3]. The HAR system infers the user's activity by reasoning over an activity model in an ontology. The products of this inference are then again saved in the real-time database so that external applications can use them. In our case, the external applications developed are the web Interface and the vocal interface. The Web interface to allow the care home to monitor multiple users at the same time in a unique and user-friendly way. In this Chapter, we won't describe in detail this component because it's not the focus of this thesis. The vocal interface is instead an application for the home user. Indeed, on the basis of the information on the database, it will initiate a conversation with the interlocutor. Sensing layer: The smart watch plays a primary role in the collection of sensory data, as shown in Figure 5.23. It deals with (i) establishing a connection with the nearest Estimote Beacon [7], via Bluetooth protocol. By placing one or more Proximity Beacons in each room of a house and knowing which room matches each beacon, it is possible to trace the user's position based on which beacon the smart watch has connected to.

This information is acquired and then published on the database by an Android application installed on the smart watch. The application pings the beacons and, depending on the Bluetooth signal strength, establishes the proximity to which beacon is closer. We set the detection range of the beacons using a configuration file saved on the database and the android application defines which beacon is nearby by publishing the numbered label corresponding to the beacon detected on the real-time database.

A second application installed on the smart watch, takes care of (ii) acquiring the inertial raw data from the smart watch worn on the user's right wrist. The application acquires 3-axis inertial data through the IMU embedded in the smart watch and sends the raw data via Wi-Fi

[7]https://estimote.com/

to the ML module. The raw sensor data is assumed to be post-processed to a point where it is a binary value with its associated label and timestamp. For instance, IMU data from a smart-watch is processed by ML module to a point where a label such as 'pouring' is true or false and has a certain timestamp. This post-processed data is then published to the database as a statement.

Human Activity Recognition layer

We take a hybrid approach towards HAR. On the one hand, adopting a machine learning approach (particularly supervised learning) for recognizing actions, e.g., (i) drinking and (ii) pouring. On the other hand, adopting an ontology based approach for recognizing contextual activity, e.g., (i) having breakfast. A real-time database sits between the machine learning module and HAR (i.e., ontology and reasoner) module, see Figure 5.23. Moreover, the database is a central module wherein all other modules save and access data. Hence, Arianna+ allows to have a hybrid approach, i.e., at the level of the system's architecture.

The machine learning module receives the IMU data stream from the smartwatch and gives as output the recognized activity. At the same moment the smartwatch sends the location of the user based on the smartwatch's proximity to the bluetooth beacons that are placed in all the house's rooms. Therefore, the recognized activity and the user location are received by the real-time database.

The real-time database in turn interacts the the HAR module. It is composed of a network of ontologies. The ontology describes the smart-home environment using axioms that are added to the ontology by hand using the Protégé [168] ontology editor, as shown in Figure 5.24. As shown in the figure, the ontology is made up of classes (ontology's T-Box), object-properties and data-properties (ontology's R-Box), and individuals (ontology's A-Box). Also for activity modeling the ontology contains SWRL rules.

After having developed the overall ontology using the Protégé editor, to be able to use the ontology as part of an overall HAR system's architecture, OWLOOP API [8] was used. The API allows to read axioms from an ontology and write axioms to it, from an OOP (object oriented programming) domain. As soon as there is new information in the real-time database, regarding the user's location and/or action, the ontology network takes this information as its input and uses that information to make some assertions in the ontology and reason based on the new assertions. The reasoner used is Pellet. If reasoning leads to the inference of a contextual activity, e.g., having breakfast, then this information is saved back in the real-time database. Lastly, when there is new information related to the recognition of the contextual

[8] https://github.com/TheEngineRoom-UniGe/OWLOOP

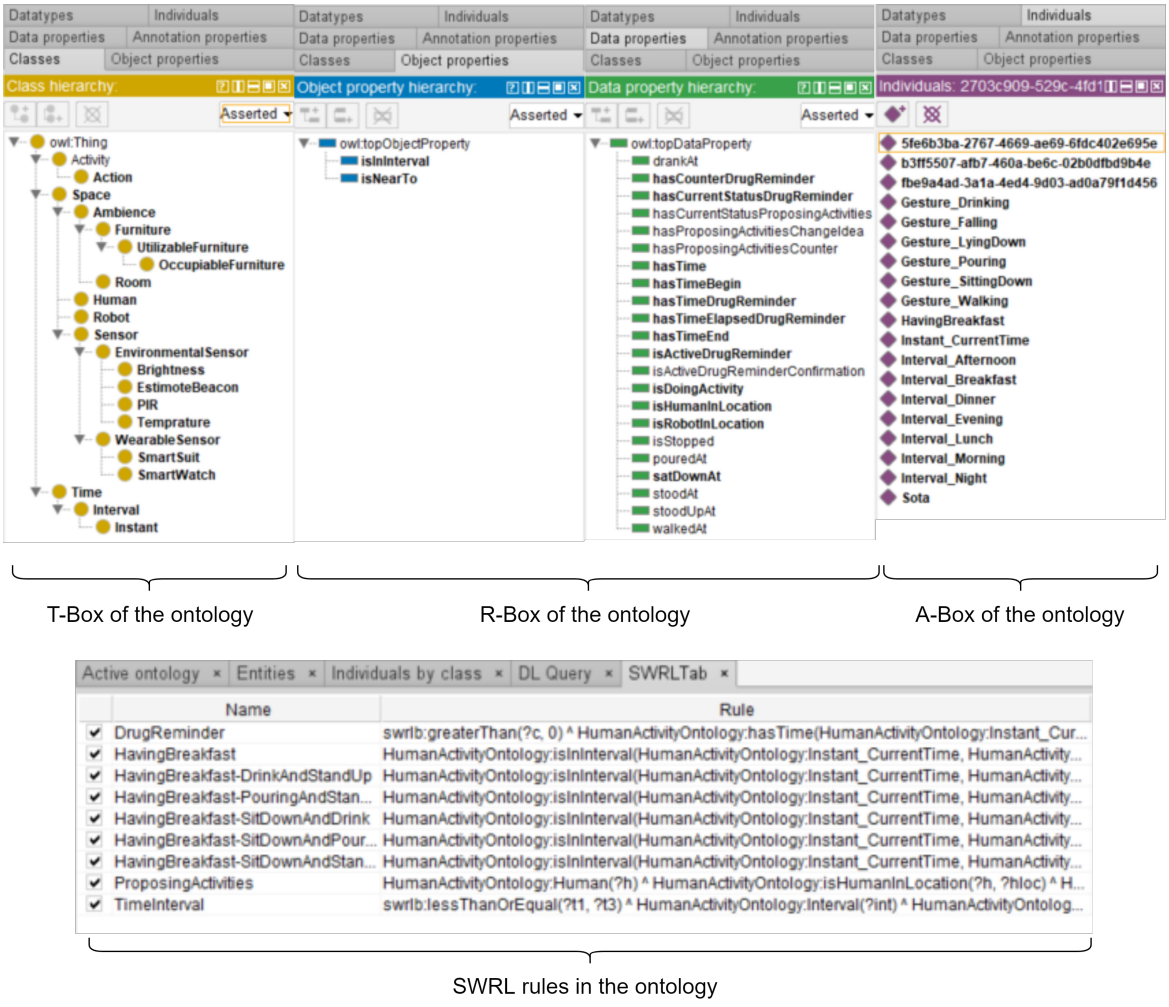


Figure 5.24 Ontology as seen in the Protégé editor.

activity (performed by the user) in the real-time database, the human computer interaction layer becomes active.

Human Computer Interaction layer

The HCI layer interacts with the user through a vocal interface. It gets triggered by the HAR layer and starts a conversation with the user. We distinguish two different user-experience of interaction. Firstly, using a smart speaker, and secondly, using a robot assistant named Sota, as shown in Figure 5.25. In practice, the HCI layer gets triggered by the contextual user activity information present in the database and manages the dialogue with the user through Google's Dialogflow wherein the logical flow of the dialogues is defined.

Dialogflow is a natural language understanding platform offered by Google that makes it easy to design and integrate a conversational user interface into our interactive voice response system. With Dialogflow we developed a conversational-agent able to interpret user's sentences and give appropriate context-based response. Indeed a typical dialogue agent has several Intents that represent a range of assisted person intentions. Whenever a user says something to Dialogflow agent, the conversational-agent attempts to match the utterance to a particular intent; then, the agent returns the response within that intent.

Sota robot Smart speaker

Figure 5.25 Figure on the left shows robot assistant (Sota) and figure on the right shows vocal assistant.

Figure 5.25 shows the robot assistant and vocal assistant. The Sota manager module as seen in Figure5.23 is connected to the Sota robot. Both the Sota manager module and the Smart speaker module are connected to the real time database and Dialogflow. Both get triggered based on the context-activity recognized in the HAR layer.

5.4.3 Experimental evaluation

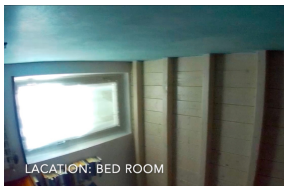
Experimental setup

The experimental setup was designed for a particular scenario that demonstrates the potential of context-based proactive human-computer interaction.

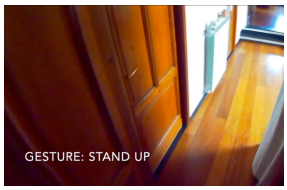
Medication reminder - scenario: Consider that the HAR system detects the presence of the user in the kitchen and recognizes that he/she is having breakfast. The HAR system has the knowledge that the user must take particular medications in the morning on full stomach. Therefore, the overall smart-home application reminds the user to take the required medications with a glass of water. To do this, the smart-home application uses the robot-assistant located on the kitchen table. It tries to recognize whether the user takes the medications or not (for instance, by detecting whether the user has accessed the medicine cabinet and recognizing whether the user does the pouring and drinking actions). After some time passes, the smart-home application inquires with the user whether he/she has taken the medications.

A close-knit interaction between different modules of the overall smart-home system makes the scenario possible. The Bluetooth proximity beacons, i.e., Estimote beacons were installed in a real home and with the support of a volunteer wearing a smart watch, an experiment was performed. Since the Estimote beacons interfere with each other, proper proximity-based user-localization requires beacons' calibration. The calibration process includes placing the beacons strategically and tuning the Bluetooth range of each beacon. The same experiment is performed with two interfaces, as described in Section 5.4.2. This allows the user to experience two kinds of interactions, i.e., an interaction with a vocal-assistant and an interaction with a robot-assistant. While the experiment was being performed the volunteer's interaction with the smart-home was recorded from an egocentric point of view (i.e., the camera is placed on the forehead of the volunteer). The volunteers interaction with both the interfaces is recorded. Therefore, we developed two videos, one showing user-interaction with the vocal assistant and another showing interaction with the robot assistant.

The current Covid-19 pandemic [137, 138] has made it difficult to test the system on a large number of volunteers. Therefore, the decision was made to video record the volunteer's experience from a ego-centric point of view to share the video with a large number of volunteers and conduct an online survey. This way, at the end of watching a video, the users were presented with the user experience questionaire (UEQ). Each user watched two



(i) User is in the bed-room



(ii) User gets out of bed

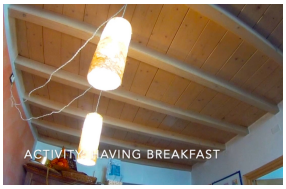
(iii) User walks

(iii) User enters in the kitchen

(v) User sits at the table

(vi) User pours water into a glass

(viii) User drinks a glass of water



(ix) User has breakfast

Figure 5.26 Different moments of the video recording from the ego-centric point of view - for medication reminder scenario.

videos and hence filled two questionnaires. One video shows the medication reminder scenario with the vocal assistant and another shows the scenario with the robot assistant. Different moments of the video recording from the ego-centric point of view are shown in the Figure 5.26. Two videos were recorded, one corresponding to the vocal [9] assistant and one for the robot assistant [10].

Experiment

The choice to video record the experience of the volunteer allowed us to reach a large number of virtual volunteers. Hence, guaranteeing us a sufficient population for our study. The questionnaire consists of five sections:

1. acquiring personal information of the interviewee such as age or experiences in a scientific/technological sector;
2. finding out if the interviewee had ever had an experience with a robot;
3. evaluating the simulated experience with the robot assistant;
4. finding out if the interviewee had ever had an experience with voice assistant;
5. evaluating the simulated experience with the vocal assistant;

[9] https://youtu.be/85BQhc87pqA
[10] https://youtu.be/w9-w5tZRZDE

The questionnaire chosen to evaluate the UX of the system equipped with the vocal assistant (section c) and system equipped with the robot assistant (section e) was created in Germany in 2005 [166]. A data analytic approach was used in order to ensure a practical relevance based on six scales where each them describes a distinct quality aspect of a system. In brainstorming sessions with usability experts, an initial item set of 229 potential items related to user experience was created. This item set was then reduced to an 80 items raw version of the questionnaire by expert evaluation. In several studies focusing on the quality of interactive products, including a statistics software package, cell phone address books, an online-collaboration software or business software, data were collected with this 80 items raw version. In total, 153 participants answered the 80 items of the raw version. Finally, the six UEQ scales and the items representing each scale were extracted from this data set by principal component analysis. The items have the form of a semantic differential, i.e. each item is represented by two terms with opposite meanings. The order of the terms is randomized per item, i.e. half of the items of a scale start with the positive term and the other half of the items start with the negative term. It was used a seven-stage scale to reduce the well-known central tendency bias for such types of items.

The UEQ contains six scales and listed below, with 26 items:

- *Attractiveness*: Overall impression of the system. Do users like or dislike the system?
- *Perspicuity*: Is it easy to get familiar with the system? Is it easy to learn how to use the system?
- *Efficiency*: Can users solve their tasks without unnecessary effort?
- *Dependability*: Does the user feel in control of the interaction?
- *Stimulation*: Is it exciting and motivating to use the system?
- *Novelty*: Is the system innovative and creative? Does the system catch the interest of users?

The consistency of the UEQ scales and their validity was investigated in 11 usability tests with a total number of 144 participants and an online survey with 722 participants. The results of these studies showed a sufficiently high scale consistency (measured by Cronbach's Alpha) [164]. In addition, the questionnaire is provided already translated and validated into 21 languages, including the languages used in this case, namely English and Italian.

The survey got compiled by 240 interviewers. But the compilations needed some filtering. As the experiment was virtual and the survey was on the internet, it is possible that not all

volunteers will answer all questions seriously. For this reason we used a simple heuristic method to detect random or not serious answer. The UEQs are composed by 24 items, 4 for each scale and our method to detect suspicious responses is based on the following questionnaire features:

- each of the 4 items associated to one scale should measure a similar User Experience (UX) quality aspect;
- each item does a qualitative evaluation on a scale from 1 to 7 but only for half of the items the number 7 corresponds to an extremely positive judgment because for the other half of the items the positive judgment corresponds to 1;

On the basis of these characteristics, for each scale, it is possible to calculate the maximum distance between the most positive and most negative judgment.

not understandable	o o o o o x o	understandable
easy to learn	o o o o o o x	difficult to learn
complicated	o o o o x o o	easy
clear	o o o o o x o	confusing

Figure 5.27 Example of the responses to the items of the scale Perspicuity.

In the Figure 5.27 is shown an example of the responses to the items of the scale Perspicuity. Obviously, these answers are not very consistent. If they are transferred to the order negative (1) to positive (7), then we can see that the ratings vary from 1 to 6, i.e. the distance between the best and worst answer is 5. Thus, a high distance between the best and the worst answer to all items in a scale is an indicator for an inconsistent or random answer behaviour.

If such a high distance occur only for a single scale this is not really a reason to exclude the answers of a participant, since such situations can also result from response errors or a simple misunderstanding of a single item. If this occurs for several scales, then it is likely that the participant has answered at least a part of the questionnaire not seriously. Thus, a simple heuristic is to consider a response as suspicious if for 3 scales the distance between best and worst response to an item in the scale exceeds 3.

In light of this criterion, we found 14 suspicious responses in the evaluation of the system equipped with a robot assistant and 13 in the evaluation of the system equipped with a vocal assistant. Once these preliminary filtering phases were completed, we proceeded to evaluate the results.

Result

Firstly, we analyzed the system equipped with the vocal assistant. In the table 5.4 are shown the means and the variance for each scale evaluated. Values between -0.8 and 0.8 represent a more or less neutral evaluation of the corresponding scale, while values > 0.8 represent a positive evaluation and values < −0.8 represent a negative evaluation.

Scale	Mean	Variance
Attractiveness	1.186	0.97
Perspicuity	1.781	1.04
Efficiency	1.302	0.92
Dependability	1.291	0.84
Stimulation	1.004	1.07
Novelty	1.040	1.09

Table 5.4 Evaluation (mean and variance) of the system equipped with Vocal assistant.

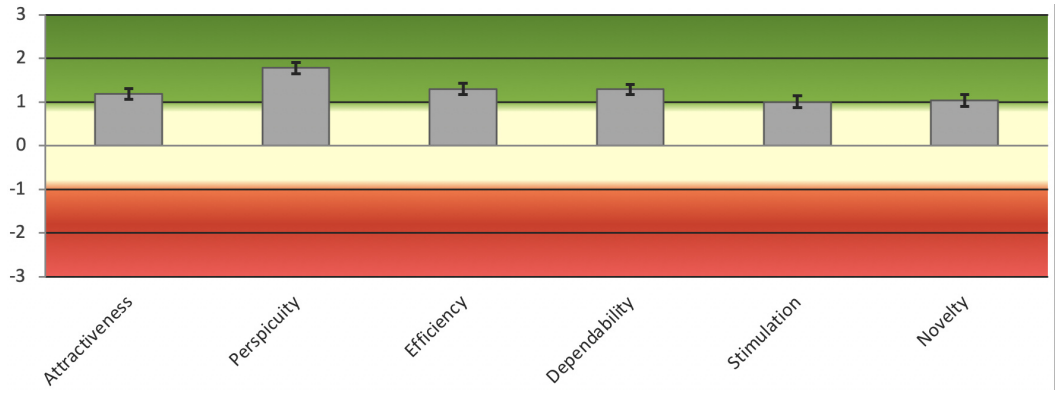


Figure 5.28 Histogram shows evaluation (mean and variance on the six scales) of the system equipped with Vocal assistant interface.

The same results presented in table 5.4 are shown in the Figure 5.28, where the range of the scales is between -3 (extremely negative) and +3 (extremely positive).

The measured scale means are set concerning existing values from a benchmark data set, which contains the data of 246 product evaluations with the UEQ (with a total of 9905 participants in all evaluations). The comparison of the results for the evaluated product with the data in the benchmark allows to draw conclusions about the relative quality of the evaluated product compared to other products.

Scale	Mean	Benchmark	Interpretation
Attractiveness	1.19	Below average	25% of results better, 50% of results worse
Perspicuity	1.78	Good	10% of results better, 75% of results worse
Efficiency	1.30	Above Average	25% of results better, 50% of results worse
Dependability	1.29	Above Average	25% of results better, 50% of results worse
Stimulation	1.00	Above Average	25% of results better, 50% of results worse
Novelty	1.04	Above Average	25% of results better, 50% of results worse

Table 5.5 Vocal assistant interface compared to benchmark.

As we can notice, the system equipped with the vocal assistant has achieved good results, or at least above average for most of the scales. Therefore, the positive results, both in the analysis of the means 5.28 and in the benchmark confirms us that the vocal assistant is positively perceived, but it leaves us with a negative result for the scale of the *attractiveness* for which it will be necessary to take action.

Secondly, the same considerations are made for the system equipped with the robot assistant. First, the means and variances of the robotic assistant system were analyzed as follows:

Scale	Mean	Variance
Attractiveness	1.440	0.95
Perspicuity	1.857	0.94
Efficiency	1.299	0.85
Dependability	1.331	0.77
Stimulation	1.320	1.11
Novelty	1.410	1.09

Table 5.6 Evaluation (mean and variance) of the system equipped with Robot assistant.

As we can see from Table 5.6 and Figure 5.29 the results are really good and very promising. Indeed, all the means are far above threshold 0.8, and we can also notice a high value of *Perspicuity* that say to us that the system is really easy, also without any explanations.

The benchmark analysis then confirmed this preliminary assessment of our means. In fact, as in the vocal assistant case 5.5, we evaluated the means in relation to existing values from a benchmark data set. The results shown in Table 5.7 are encouraging because all means

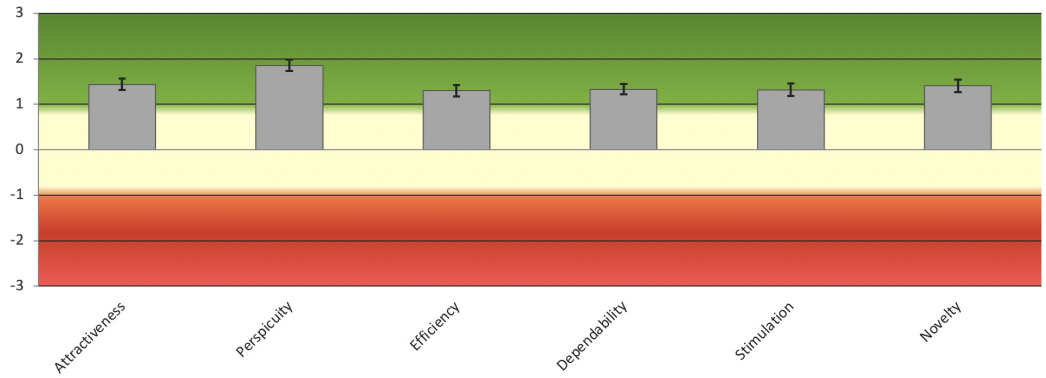


Figure 5.29 Histogram shows evaluation (mean and variance on the six scales) of the system equipped with Robot assistant interface.

are above average and two of the six scales are even in the range of 10% of the best results in the reference data set. Indeed, these results tell us that, even more markedly than the vocal assistant system, it is a system that is really simple to use and very innovative.

Scale	Mean	Comparison to benchmark	Interpretation
Attractiveness	1.44	Above average	25% of results better, 50% of results worse
Perspicuity	1.86	Excellent	In the range of the 10% best results
Efficiency	1.30	Above Average	25% of results better, 50% of results worse
Dependability	1.33	Above Average	25% of results better, 50% of results worse
Stimulation	1.32	Above Average	25% of results better, 50% of results worse
Novelty	1.41	Excellent	In the range of the 10% best results

Table 5.7 Robot assistant interface compared to benchmark..

The two systems have therefore met with some success among the interviewees. Therefore we also decided to evaluate which of the two systems can be considered the preferred one. Already preliminary, we noticed higher means in the robotic system, but to verify they really preferred the robotic system, we compared the two systems' results.

	Vocal assistant		Robot assistant	
Scale	Mean	Std. Dev.	Mean	Std. Dev.
Attractiveness	1,19	0,99	1,44	0,97
Perspicuity	1,78	1,02	1,86	0.97
Efficiency	1,30	0,96	1,30	0.92

Dependability	1,29	0,92	1,33	0.88
Stimulation	1,00	1,03	1,32	1.05
Novelty	1,04	1,04	1,41	1.04

Table 5.8 Comparison between the interaction with Vocal assistant and Robot assistant.

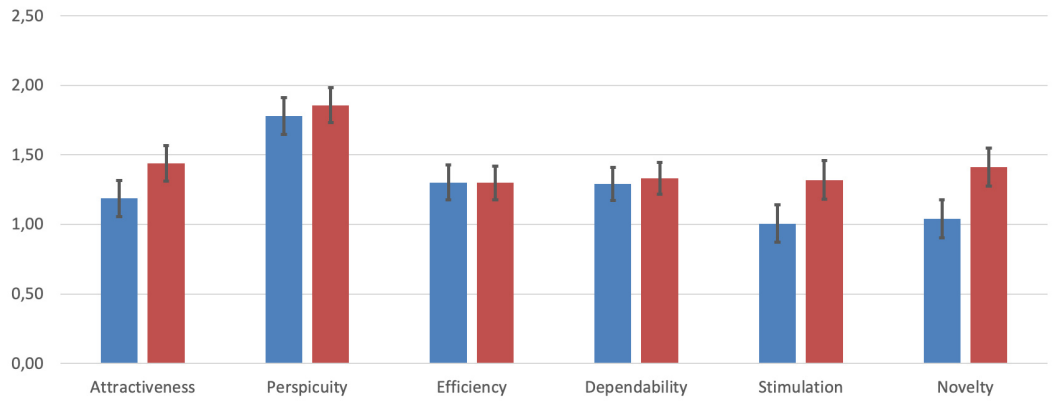


Figure 5.30 Vocal assistant system Robot assistant system

From Figure 5.30 as from table 5.8, it is evident that the interaction with a robot assistant always has a higher mean. This leads us to think that the interviewees actually prefer the interaction with a robot assistant rather than the interaction with a vocal assistant. This is further confirmed by the T-Test. From the T-Test (table 5.9) we can, in fact, note that the interaction with a robot assistant is significantly better in 3 of the 6 scales of interest. The T-Test is used to check if the mean values of products is significantly different. As suggested in the literature [169], a p-value of less than 0.05 suggests significant difference. From this, we can conclude with certainty that the robotic system is more attractive, stimulating and innovative than the interaction with a vocal assistant.

Scale	p-value	Difference
Attractiveness	0.0061	Significant Difference
Perspicuity	0.4137	No Significant Difference
Efficiency	0.9721	No Significant Difference
Dependability	0.6356	No Significant Difference
Stimulation	0.0014	Significant Difference
Novelty	0.0002	Significant Difference

Table 5.9 T-Test comparison between the interaction with Vocal assistant and Robot assistant.

Based on results shown in the Evaluation Section, and based on literature review, this research work propose proposes an alternative vision to the classic smart home systems for monitoring human activities, showing how the integration of user-friendly interface elements such as voice assistants or robotic assistants can guarantee a positive perception of the system. In fact, what emerges from the results is that (i) in terms of functionality both robot assistant and vocal assistant have been perceived equally well (Fig. 5.5, Fig. 5.7), (ii) whereas, in terms of the appeal, the robot assistant is perceived as better compared to vocal assistant (Fig. 5.30). We propose that, the scales *perspicuity, efficiency, dependability* capture the perception of the 'functionality' of the system. The 'functionality' is the contextual and proactive interaction provided by the digital assistant. The scales *attractiveness, stimulation, novelty* capture the perception of the 'appeal' of the system. The 'appeal' is affected by the morphology of the digital assistant. In summary both interfaces are perceived positively by the users but the robot assistant is perceived better on the scale of attractiveness, stimulation and innovation.

5.4.4 Summary

This work presents a HAR system based on Arianna+ and developed using OWLOOP API. It has been integrated with a dialogue management system and a digital assistant. This system presents a scenario wherein, based on the HAR results, a digital assistant provides contextual and proactive support to the home user. In this scenario, once the system recognizes that the user is having breakfast, it triggers the dialogue management system and the digital assistant, which thus reminds the user to take medication. With respect to the motivations presented in Section1.1.1, this work highlights that robust HAR results can be useful for smart-home applications. This is particular work, the HAR results provide contextual information to a digital assistant. Apart from the original motivation, in this work, we evaluate a user's experiences while interacting with two different kinds of digital assistants, i.e., one with a robot body and one without a robot body.

To evaluate the user experience with a large number of volunteers, we video-recorded an experiment from an ego-centric point of view. Volunteers were then requested to watch the video, experience the scenario virtually and answer the user experience questionnaire (UEQ). Results of the survey, show that the digital assistant with a robot body is comparatively perceived as more attractive, stimulating, and novel. With this preliminary result, we argue that a smart-home application - such as a companion robot assistant providing contextualized and proactive support to the home-users - could positively affect the acceptability of a smart-home. The volunteers' experiencing of the digital assistants virtually, i.e., by watching a video, could be considered as a limitation of this work. Nonetheless, this is one way to perform experiments when it is difficult for a large number of volunteers to experience the real-world setup due to the Covid-19 pandemic [137, 138]. In the future, we intend to perform the same evaluation in a real-world setup with more number of contextualized and proactive support scenarios. Furthermore, with digital-assistants placed in multiple rooms of the apartment.

Chapter 6

Conclusion and future work

6.1 Conclusion

Human activity recognition (HAR) has been a very active research topic in the past two decades for its applications in various fields such as health-care, security and surveillance, comfort and safety, and energy consumption. The work of this thesis particularly finds its motivation in the health-care domain. It is estimated that there will be an increase in the size of the older population between 2020 and 2050. Globally, the share of the population aged 65 years or over is expected to increase from 9.3 percent in 2020 to around 16.0 percent in 2050 [1]. The same report highlights that for older persons to live independently, some universal needs must be met, which include health care services. This particular point is not new and has been relevant for many years. Furthermore, it has been one of the motivations for smart home research - particularly in the past decade - due to advances in sensing, networking, and ambient intelligence technologies. To support healthy and active aging, to maintain and improve the quality of life of older persons, and to respond to the needs of the rapidly aging population - researchers are developing smart-home environments that can recognize ADL. The ability to perform ADL without assistance from other people can be considered as a reference for the estimation of the independent living level of the older person. Conventionally, this has been assessed by health-care domain experts via a qualitative evaluation of the ADL. Since this evaluation is qualitative, it can vary based on the person being monitored and the caregiver's experience. A significant amount of research work is implicitly or explicitly aimed at augmenting the health-care domain expert's qualitative evaluation with quantitative data or knowledge obtained from HAR.

From a medical perspective, there is a lack of evidence about the technology readiness level of smart home architectures supporting older persons by recognizing ADL [2]. We

hypothesize that this may be due to a lack of effective collaboration between smart-home researchers/developers and health-care domain experts, especially when considering HAR. We foresee an increase in HAR systems being developed in close collaboration with caregivers and geriatricians to support their qualitative evaluation of ADL with explainable quantitative outcomes of the HAR systems. This has been a motivation for the work in this thesis. The recognition of human activities – in particular ADL – may not only be limited to support the health and well-being of older people. It can be relevant to home users in general. For instance, HAR could support digital assistants or companion robots to provide contextually relevant and proactive support to the home users, whether young adults or old. This has also been a motivation for the work in this thesis.

Depending on the application or motivation, researchers would like to have a certain set of features in the HAR systems they develop. In the literature, one can see researchers attempting to incorporate features such as (i) scalability of the HAR system in terms of accommodating more number of heterogeneous sensors and more variety of activities that can be recognized, (ii) online activity recognition which means computational performance must be taken into consideration, (iii) accuracy of activity recognition, part of which is also the ability to deal with uncertainty-of and noise-in sensory data, (iv) intelligibility in terms of explainable results and inner-workings of the system, (v) learning ability in order to improve HAR accuracy over time, and (vi) consideration to privacy. It is noteworthy that within the literature each feature is a research problem [6]. In terms of categorizing HAR approaches, the literature can be divided based on sensing approaches and modeling approaches. On the one hand, sensing approaches are divided into visual sensor-based and non-visual sensor-based. On the other hand, modeling approaches are divided into data-driven, knowledge-based and hybrid. For HAR within smart-homes, most research work has used a non-visual sensor-based approach. Although cameras provide high accuracy for activity or action recognition, within a smart home environment due to privacy issues simpler sensors (e.g., Passive Infrared (PIR), light, RFID, wearables inertial sensors, etc.) are largely used. In terms of modeling approaches, there has been rich and extensive research in data-driven and knowledge-based approaches, and in the past few years very active research in hybrid approaches. Most research that is related to hybrid modeling approaches have algorithmic orientation and some research is oriented towards hybrid frameworks and architectures.

Given our motivations, namely, (i) facilitation of iterative development and ease in collaboration between HAR system researchers/developers and health-care domain experts in ADL, and (ii) robust HAR that can support digital assistants or companion robots. There is a need

for the development of a HAR framework that at its core is modular and flexible to facilitate an iterative development process [3], which is an integral part of collaborative work that involves develop-test-improve phases. At the same time, the framework should be intelligible for the sake of enriched collaboration with health-care domain experts. Furthermore, it should be scalable, online, and accurate for having robust HAR, which can enable many smart-home applications. The goal of this thesis is to design and evaluate such a framework.

The contributions of this thesis are divided into three parts. The first contribution is Arianna+, a framework to develop networks of ontologies - for knowledge representation and reasoning - that enables smart homes to perform human activity recognition online. The second contribution is OWLOOP, an API that supports the development of HAR system architectures based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object-Oriented Programming (OOP). The third contribution is the evaluation and exploitation of Arianna+ using OWLOOP API. The exploitation of Arianna+ using OWLOOP API has resulted in four HAR system implementations. The evaluations and results of these HAR systems emphasize the novelty of Arianna+. This thesis is structured in the following way. There are six chapters in total. Chapter 1 gives an introduction to the domain of HAR, it presents the motivations for this thesis, the challenges and the problem description. Chapter 2 presents the state of the art in HAR aimed at health and well-being in smart-homes. Based on the literature review, this chapter also presents the research questions addressed by this thesis. Chapter 3 presents the first contribution of this thesis, i.e. Arianna+. Chapter 4 presents the second contribution of this thesis, i.e., OWLOOP API. Chapter 5 presents the third contribution of this thesis, i.e., HAR system implementations based on Arianna+ and developed using OWLOOP API, whose evaluation and results address the research problems highlighted in Section 2.4.

The first contribution - Chapter 3 presents Arianna+, a framework for HAR that adopts the nodes-and-edges design. Where the nodes are ontologies and edges are computational procedures. The ontologies are *a priori* defined contextual knowledge structures that can be updated with statements (i.e., axioms) based on sensor data. The computational procedures transfer or generate statements. A transfer of statements takes place from one ontology to another based on pre-defined events and conditions. New statements are generated by reasoning on the fluent models (i.e., rules) present in the ontologies. These statements are mapped from event-based sensor data. In order to exploit sensors providing a continuous data stream (e.g., inertial data or video) computational procedures can accommodate data-driven techniques, which can process the continuous data stream and provide an event as a result. Thus, in this manner, an ontology network is intrinsically *modular* due to its nodes-and-edges

design and *intelligible* due to its knowledge-based approach. The ability to accommodate both knowledge-based and data-driven techniques for the generation of statements (i.e., by reasoning on activity models) makes Arianna+ architecturally hybrid. Moreover, also *scalable* in terms of being able to accommodate more activity models within the network and data from heterogeneous sensors. This is because the activity models could be knowledge-based or data-driven hence they can deal with heterogeneous sensor data (i.e., event-based like data from PIR, pressure-mats, etc., or continuous-stream like data from accelerometers, gyroscopes, videos, etc.). Arianna+ is designed for *online* HAR. Although it is known that OWL-based reasoning is exponentially complex with respect to the number of axioms in the ontology [77], Arianna+ deals with this issue as it distributes knowledge over a network of ontologies and reasons only on contextualized relevant knowledge. Arianna+ is designed for *accurate* HAR. Due to its modularity and scalability, one could design a redundant network, wherein multiple activity models (using different approaches) can be used to recognize the same activity and the most accurate HAR result can be selected based on a heuristic. Considering all these aspects Arianna+ attempts to address all research questions presented in Section 2.4, i.e., **RQ1, RQ2, RQ3, RQ4, RQ5**.

The second contribution - Chapter 4 presents OWLOOP API that is designed to support and ease the development of HAR systems based on Arianna+. It enables the usage of Ontology Web Language (OWL) by the means of Object-Oriented Programming (OOP). OWLOOP API performs a passive OWL to OOP mapping that allows to manipulate knowledge in ontology files and perform inference by using OWL reasoners. OWLOOP allows to construct and use Descriptors, which are Java classes that interface OOP objects with knowledge structured in program-memory as an ontology. Descriptors encapsulate boiler-plate code to simplify the development and maintenance of a system that exploits knowledge representation and reasoning using ontologies. The Descriptors' methods allow to read, write, update, delete, and reason on axioms in ontology files. Furthermore, flexibility in Descriptor construction allows avoiding drawbacks in computational performance. OWLOOP is suitable for complex applications requiring the management of dynamic ontologies. With all these aspects OWLOOP API attempts to address research questions **RQ1**, **RQ2**, and **RQ4** presented in Section 2.4.

The third contribution - Chapter 5 presents in its four sections HAR systems based on Arianna+ and developed using OWLOOP API. Their evaluation and results particularly highlight the novelties of Arianna+ and the research questions they address in this thesis.

Section 5.1 presents a HAR system whose objective was to perform a preliminary evaluation of the computational performance while reasoning using a network of ontologies.

An experimental setup was designed with a use case scenario comprising of five activity models. Experiments were performed using a simulated dataset for evaluating the behavior of the network and its computational performance. As the complexity of the system increases (i.e., number of axioms in the system) the overall reasoning time is less for the case of CAE (contextualized activity evaluation) as compared to the case of PAE (parallel activity evaluation). However, we restrain from making conclusive statements about the comparison, because of the presence of high variability in the result of the CAE case (as can be seen in Figure 5.5). The evaluation of this work addresses the research questions **RQ2** and **RQ5** presented in Section 2.4.

Section 5.2 presents a HAR system whose objective was to evaluate a HAR system using the CASAS dataset. For this system, an ontology network is developed wherein there are nine ontologies. One ontology called the `place ontology` is used to spatially relate sensory data and trigger (based on events) temporal reasoning on one or more of the remaining eight activity ontologies. Hence, the reasoning is concurrent and based on a hierarchy of contexts, which are represented as small ontologies. Due to contextualized reasoning computations are performed on minimum relevant data and only when required. Due to these reasons, activity recognition is online. Furthermore, this work highlights intelligibility in the inner-workings of the system by fluent models of all the eight activities and their representation in natural language. This emphasizes the point that modular and intelligible systems can enable ease in an iterative development process in collaboration with health-care domain experts in ADL. The evaluation of this work addresses the research questions **RQ1**, **RQ2**, and **RQ5** presented in Section 2.4.

Section 5.3 presents a HAR system whose objective was to showcase a prototype real-world HAR system that saves its activity recognition results in a purpose-built query ontology. SPARQL can then be used to query this ontology to retrieve results in an intelligible manner. Explainable results are particularly relevant for an enriched collaboration between HAR system researchers/developers and health-care domain experts. This work presents the SPARQL queries, their meaning in natural language, and their intelligible results. The evaluation of this work addresses the research questions **RQ1**, **RQ3**, and **RQ4** presented in Section 2.4.

Finally, section 5.4 presents a HAR system whose objective was to integrate HAR results with a dialogue management system and a digital assistant. This system presents a scenario wherein, based on the HAR results, a digital assistant provides contextual and proactive support to the home user. In this scenario, once the system recognizes that the user is having breakfast, it triggers the dialogue management system and the digital assistant, which

thus reminds the user to take medication. This work highlights that robust HAR results can be useful for smart-home applications. Particularly in this work, the results provide contextual information to a digital assistant. Apart from the original motivation, in this work, we evaluate a user's experiences while interacting with two different kinds of digital assistants, i.e., one with a robot body and one without a robot body. Volunteers experience the experiment virtually and complete a survey called user experience questionnaire (UEQ). Results of the survey, show that the digital assistant with a robot body is comparatively perceived as more attractive, stimulating, and novel. In relation to the integration of the HAR system with the dialogue management system and digital assistant, the evaluation of this work addresses the research questions **RQ2**, **RQ3**, and **RQ4** presented in Section 2.4.

6.2 Future work

Human activity recognition is an active and challenging research domain. Particularly with respect to the development of hybrid solutions for HAR. The evaluations and preliminary results of this thesis are encouraging, and we plan to further improve upon our contributions by overcoming limitations in the present work and further investigating several interesting research directions. Hence, with respect to the overall thesis conclusion presented in the previous section, we highlight limitations of our work and interesting research directions.

Arianna+ is a modular and intelligible framework for developing HAR systems with key features such as scalability, and online and accurate HAR. But a limitation of Arianna+ is the requirement of an extensive knowledge engineering effort while developing HAR systems. Hence, an interesting research direction to explore would be to organize all existing upper ontologies that can be reused to reduce the knowledge engineering effort. This exploration would also require careful thought over the relevancy of various upper ontologies. Another limitation of the framework is that the reasoning is deterministic. The framework in its current state (i.e., with knowledge based on DL, and reasoning based on rules that are deterministic) is not robust to missing or noisy sensor data. This limitation can be overcome by further developing the framework to easily accommodate reasoning using fuzzy[81] or probabilistic [82] OWL reasoners. Apart from these aspects, an interesting research direction would be to explore how active learning can be used to fine-tune or improve existing knowledge-based activity models in the system.

OWLOOP API eases the development of HAR systems that are based on Arianna+. The current version of the OWLOOP API concerns only with the expressions shown in Table 4.3, which do not encompass all the OWL axioms. For instance, it does not allow the

representation of classes through structured disjunctions and conjunctions of class restrictions as defined by OWL. Instead, it considers all the restrictions to be in conjunction with each other without a specific order. For most applications, this is not a limitation because it is possible to use an ontology editor (*e.g.*, Protégé [127]) to design static semantics. Then, the ontology can be loaded and subjected to runtime operations through OWLOOP descriptors. In the future, OWLOOP can be modularly extended to support all OWL axioms by designing new types of `OWLOOPEntity` and related `Descriptors`. Current limitations of OWLOOP are the following: (i) the effect on the computational performance (due to the overhead by the use of Descriptors, which are encapsulating Java-interfaces, classes, and methods in the OWL API) has not been quantitatively measured and (ii) based on the API design we hypothesize that flexibility in Descriptor construction allows to avoid drawbacks in computational performance - this also remains to be quantitatively confirmed. Nonetheless, these limitations can be overcome with further research into the API and they do not hinder the practical use of the API while building HAR systems - as could be seen in Chapter 5.

HAR system presented in Section 5.1 shows results of the comparison made between contextualized activity evaluation (CAE case) and parallel activity evaluation (PAE case). Results with CAE (contextualized activity evaluation) indicate that a HAR system that performs contextualized-reasoning using a network of ontologies has better computational performance as compared to a HAR system that performs reasoning in a non-contextualized manner, i.e., in case of PAE (parallel activity evaluation), wherein all activities are evaluated together at the same time. As the complexity of the system increases (i.e., number of axioms in the system) the overall reasoning time is less for the case of CAE as compared to the case of PAE. However, we restrain from making conclusive statements about the comparison, because of the presence of high variability in the result of the CAE case (as can be seen in Figure 5.5). Hence, future work could involve testing long-term computational performance with data arriving from a real-world smart-home setup.

HAR system presented in Sections 5.1 and 5.2 does not consider the challenging issues related to multi-occupancy. These systems are developed by considering that there is a single person in the smart home. To be able to perform activity recognition with multiple persons, the system would require sensory data tagged with a person's unique identification. Hence, systems presented in Sections 5.3 and 5.4 make use of wrist-worn smart-watches and Bluetooth beacons placed in the environment. The smart-watches are able to localize each person in the house and perform activity recognition of each unique person. But the current implementations of the two systems were only tested with two occupants in the smart-home. We speculate that as the number of occupants increases, if the same ontology

network is used for all the occupants then the complexity of knowledge in the ontology network increases, which might affect the computational performance and thus online HAR. Therefore, at the level of system development, an interesting research direction to explore would be to instantiate a new ontology network for each occupant in the house, such that these ontology networks are being evaluated in parallel. This research direction may also require further improvements in OWLOOP API and extensive testing.

Currently, all HAR systems presented in Chapter 5 are not robust to missing or noisy input data. To circumvent this limitation a relatively easy solution would be to incorporate redundant activity models, i.e., multiple activity models using different techniques to recognize the same activity and based on a predefined heuristic the most accurate result (between multiple activity models recognizing the same activity) can be considered as the true one. Another solution could be further development of the framework to easily accommodate reasoning using fuzzy[81] or probabilistic [82] OWL reasoners. Nonetheless, Section 5.3 shows a HAR system that gives good HAR results for the CASAS dataset. Although the results of our approach are compared with other state of the art hybrid approaches (see Table 5.3), we restrain from making any conclusive statements about the comparison, because in terms of the approach, all the state of the art approaches are algorithmically hybrid whereas our approach is hybrid in an architectural-sense. Furthermore, the fluent models (i.e., activity models) in our approach are hand-crafted and the activity models in the state of the art approaches are learnt from the dataset. This highlights that Arianna+ framework being hybrid in an architectural-sense is a novel approach.

The Covid-19 pandemic [137, 138] also had some affect on the planned research activities. For instance, on the one hand, HAR systems presented in Sections 5.2 and 5.3 highlight the intelligibility of the fluent models and activity recognition results in natural language. But the intelligibility feature of the framework could not be evaluated by discussions with health-care domain experts as their availability was affected by the Covid-19 pandemic. On the other hand, HAR system presented in Section 5.4 shows that contextualized HAR provides relevant knowledge to companion digital assistants. This system was evaluated by providing a virtual experience (of interaction with the digital assistants) to the volunteers - as it would be difficult for large number of volunteers to experience the real-world setup due to the pandemic restrictions.

Another relevant research direction would be to explore better representations of an ontology wherein HAR results are saved, such that the SPARQL queries and their results are self-explanatory. This is particularly relevant for an enriched collaboration between HAR system researchers/developers and health-care domain experts (see Section 5.3). Finally,

two HAR systems (see Sections 5.1 and 5.2) presented in this work used datasets for their evaluation, and the remaining two HAR systems (see Sections 5.3 and 5.4) were deployed in a real-world setup but only preliminary tests were performed. Therefore, there is a need for real-world long-term evaluation of a HAR system based on Arianna+ and developed using OWLOOP API.

Lastly, as part of this thesis work, we also conducted preliminary exploration in a unique research direction. By foreseeing a future wherein there exist complex system architectures, particularly in the growing field of smart-homes, we posited a research question: *"can tools of network science be used to analyze complex system architectures? If yes, then what exactly could be the advantages of such an analysis?"*. Preliminary research allows us to present a hypothesis that *smart-home systems considered as multi-agent systems can be analyzed using tools of network science to gain insights into their complexity, robustness and vulnerability, and modularity*. This hypothesis needs to be critically tested and to the best of our knowledge, it paves a unique research path. The path can be described as - adaptation of network science methodologies to the smart-home system's literature. The preliminary research in this direction is presented in the Appendix-A of this thesis.

References

[1] Y. Kamiya, N. Mun Sim Lai, and K. Schmid, "World population ageing 2020 highlights," *Department of Economic and Social Affairs, United Nations*, pp. 1–2, 2020.

[2] L. Liu, E. Stroulia, I. Nikolaidis, A. Miguel-Cruz, and A. R. Rincon, "Smart homes and home health monitoring technologies for older adults: A systematic review," *International Journal of Medical Informatics*, vol. 91, pp. 44–59, 2016.

[3] C. Larman, *Agile and iterative development: a manager's guide*. Addison-Wesley Professional, 2004.

[4] J. Yang, J. Lee, and J. Choi, "Activity recognition based on rfid object usage for smart mobile devices," *Journal of Computer Science and Technology*, vol. 26, no. 2, pp. 239–246, 2011.

[5] G. Okeyo, L. Chen, and H. Wang, "Combining ontological and temporal formalisms for composite activity modelling and recognition in smart homes," *Future Generation Computer Systems*, vol. 39, pp. 29–43, 2014.

[6] Q. Ni, A. B. Garcia Hernando, D. la Cruz, and I. Pau, "The elderly's independent living in smart homes: A characterization of activities and sensing infrastructure survey to facilitate services development," *Sensors*, vol. 15, no. 5, pp. 11 312–11 362, 2015.

[7] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 790–808, 2012.

[8] A. Costa, V. Julián, and P. Novais, "Advances and trends for the development of ambient-assisted living platforms," *Expert Systems*, vol. 34, no. 2, p. e12163, 2017.

[9] E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 636–654.

[10] M. Khan, B. N. Silva, and K. Han, "Internet of things based energy aware smart home control system," *IEEE Access*, vol. 4, pp. 7556–7566, 2016.

[11] S. Greene, H. Thapliyal, and D. Carpenter, "Iot-based fall detection for smart home environments," in *2016 IEEE international symposium on nanoelectronic and information systems (iNIS)*. IEEE, 2016, pp. 23–28.

[12] A. Anvari-Moghaddam, H. Monsef, and A. Rahimi-Kian, "Optimal smart home energy management considering energy saving and a comfortable lifestyle," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 324–332, 2014.

[13] S. Katz, A. B. Ford, R. W. Moskowitz, B. A. Jackson, and M. W. Jaffe, "Studies of illness in the aged: the index of adl: a standardized measure of biological and psychosocial function," *Jama*, vol. 185, no. 12, pp. 914–919, 1963.

[14] G. Demiris, B. K. Hensel, M. Skubic, and M. Rantz, "Senior residents' perceived need of and preferences for" smart home" sensor technologies," *International journal of technology assessment in health care*, vol. 24, no. 1, p. 120, 2008.

[15] S. Ranasinghe, F. Al Machot, and H. C. Mayr, "A review on applications of activity recognition systems with regard to performance and evaluation," *International Journal of Distributed Sensor Networks*, vol. 12, no. 8, p. 1550147716665520, 2016.

[16] A. K. Dey, "Understanding and using context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, 2001.

[17] L. Sanchez, J. Lanza, R. Olsen, M. Bauer, and M. Girod-Genet, "A generic context management framework for personal networking environments," in *2006 Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services*. IEEE, 2006, pp. 1–8.

[18] M. Pantic, A. Pentland, A. Nijholt, and T. S. Huang, "Human computing and machine understanding of human behavior: A survey," in *Artifical intelligence for human computing*. Springer, 2007, pp. 47–71.

[19] X. Xu, J. Tang, X. Zhang, X. Liu, H. Zhang, and Y. Qiu, "Exploring techniques for vision based human activity recognition: Methods, systems, and evaluation," *sensors*, vol. 13, no. 2, pp. 1635–1650, 2013.

[20] Y. Yan, E. Ricci, G. Liu, and N. Sebe, "Egocentric daily activity recognition via multitask clustering," *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 2984–2995, 2015.

[21] S. Chernbumroong, S. Cang, A. Atkins, and H. Yu, "Elderly activities recognition and classification for applications in assisted living," *Expert Systems with Applications*, vol. 40, no. 5, pp. 1662–1674, 2013.

[22] K.-C. Liu, C.-Y. Yen, L.-H. Chang, C.-Y. Hsieh, and C.-T. Chan, "Wearable sensor-based activity recognition for housekeeping task," in *2017 IEEE 14th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*. IEEE, 2017, pp. 67–70.

[23] M. Stikic, T. Huynh, K. Van Laerhoven, and B. Schiele, "Adl recognition based on the combination of rfid and accelerometer sensing," in *2008 second international conference on pervasive computing technologies for healthcare*. IEEE, 2008, pp. 258–263.

[24] A. Hein and T. Kirste, "A hybrid approach for recognizing adls and care activities using inertial sensors and rfid," in *International Conference on Universal Access in Human-Computer Interaction*. Springer, 2009, pp. 178–188.

[25] E. Hoque and J. Stankovic, "Aalo: Activity recognition in smart homes using active learning in the presence of overlapped activities," in *2012 6th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*. IEEE, 2012, pp. 139–146.

[26] G. A. Oguntala, R. A. Abd-Alhameed, N. T. Ali, Y.-F. Hu, J. M. Noras, N. N. Eya, I. Elfergani, and J. Rodriguez, "Smartwall: Novel rfid-enabled ambient human activity recognition using machine learning for unobtrusive health monitoring," *IEEE Access*, vol. 7, pp. 68 022–68 033, 2019.

[27] J. Lee and N. Melo, "Habit representation based on activity recognition," *Sensors*, vol. 20, no. 7, p. 1928, 2020.

[28] P. Rashidi and A. Mihailidis, "A survey on ambient-assisted living tools for older adults," *IEEE journal of biomedical and health informatics*, vol. 17, no. 3, pp. 579–590, 2012.

[29] A. Queirós, A. Dias, A. G. Silva, and N. P. Rocha, "Ambient assisted living and health-related outcomes—a systematic literature review," in *Informatics*, vol. 4, no. 3. Multidisciplinary Digital Publishing Institute, 2017, p. 19.

[30] Z. Hussain, S. Sagar, W. E. Zhang, and Q. Z. Sheng, "A cost-effective and non-invasive system for sleep and vital signs monitoring using passive rfid tags," in *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2019, pp. 153–161.

[31] H. Mshali, T. Lemlouma, M. Moloney, and D. Magoni, "A survey on health monitoring systems for health smart homes," *International Journal of Industrial Ergonomics*, vol. 66, pp. 26–56, 2018.

[32] B. Bruno, J. Grosinger, F. Mastrogiovanni, F. Pecora, A. Saffiotti, S. Sathyakeerthy, and A. Sgorbissa, "Multi-modal sensing for human activity recognition," in *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2015, pp. 594–600.

[33] G. Anitha and S. B. Priya, "Posture based health monitoring and unusual behavior recognition system for elderly using dynamic bayesian network," *Cluster Computing*, vol. 22, no. 6, pp. 13 583–13 590, 2019.

[34] C. Zhu and W. Sheng, "Wearable sensor-based hand gesture and daily activity recognition for robot-assisted living," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 41, no. 3, pp. 569–573, 2011.

[35] M. A. Soliman and M. Alrashed, "An rfid based activity of daily living for elderly with alzheimer's," *Internet of Things (IoT) Technol. HealthCare*, vol. 54, 2018.

[36] M. Uddin, W. Khaksar, J. Torresen *et al.*, "Ambient sensors for elderly care and independent living: a survey," *Sensors*, vol. 18, no. 7, p. 2027, 2018.

[37] M. Vrigkas, C. Nikou, and I. A. Kakadiaris, "A review of human activity recognition methods," *Frontiers in Robotics and AI*, vol. 2, p. 28, 2015.

[38] S. Herath, M. Harandi, and F. Porikli, "Going deeper into action recognition: A survey," *Image and vision computing*, vol. 60, pp. 4–21, 2017.

[39] Y. Maret, D. Oberson, and M. Gavrilova, "Real-time embedded system for gesture recognition," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2018, pp. 30–34.

[40] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *International conference on pervasive computing*. Springer, 2004, pp. 1–17.

[41] N. Kern, B. Schiele, H. Junker, P. Lukowicz, and G. Tröster, "Wearable sensing to annotate meeting recordings," *Personal and Ubiquitous Computing*, vol. 7, no. 5, pp. 263–274, 2003.

[42] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, pp. 1–43, 2011.

[43] M. Ruzzon, A. Carfì, T. Ishikawa, F. Mastrogiovanni, and T. Murakami, "A multisensory dataset for the activities of daily living," *Data in brief*, vol. 32, p. 106122, 2020.

[44] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[45] Q. Wang, A. Timmermans, W. Chen, J. Jia, L. Ding, L. Xiong, J. Rong, and P. Markopoulos, "Stroke patients' acceptance of a smart garment for supporting upper extremity rehabilitation," *IEEE journal of translational engineering in health and medicine*, vol. 6, pp. 1–9, 2018.

[46] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, and D. Hahnel, "Inferring activities from interactions with objects," *IEEE pervasive computing*, vol. 3, no. 4, pp. 50–57, 2004.

[47] K. P. Fishkin, M. Philipose, and A. Rea, "Hands-on rfid: Wireless wearables for detecting use of objects," in *Ninth IEEE International Symposium on Wearable Computers (ISWC'05)*. IEEE, 2005, pp. 38–41.

[48] D. J. Patterson, D. Fox, H. Kautz, and M. Philipose, "Fine-grained activity recognition by aggregating abstract object usage," in *Ninth IEEE International Symposium on Wearable Computers (ISWC'05)*. IEEE, 2005, pp. 44–51.

[49] M. Buettner, R. Prasad, M. Philipose, and D. Wetherall, "Recognizing daily activities with rfid-based sensors," in *Proceedings of the 11th international conference on Ubiquitous computing*, 2009, pp. 51–60.

[50] T. Gu, Z. Wu, X. Tao, H. K. Pung, and J. Lu, "epsicar: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition," in *2009 IEEE International Conference on Pervasive Computing and Communications*. IEEE, 2009, pp. 1–9.

[51] J. Ye, S. Dobson, and S. McKeever, "Situation identification techniques in pervasive computing: A review," *Pervasive and mobile computing*, vol. 8, no. 1, pp. 36–66, 2012.

[52] A. Carfi, C. Motolese, B. Bruno, and F. Mastrogiovanni, "Online human gesture recognition using recurrent neural networks and wearable sensors," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2018, pp. 188–195.

[53] A. Fleury, M. Vacher, and N. Noury, "Svm-based multimodal classification of activities of daily living in health smart homes: sensors, algorithms, and first experimental results," *IEEE transactions on information technology in biomedicine*, vol. 14, no. 2, pp. 274–283, 2009.

[54] R. Begg and R. Hassan, "Artificial neural networks in smart homes," in *Designing smart homes*. Springer, 2006, pp. 146–164.

[55] A. Veeraraghavan, A. K. Roy-Chowdhury, and R. Chellappa, "Matching shape sequences in video with applications in human movement analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1896–1909, 2005.

[56] L. Atallah, B. Lo, R. Ali, R. King, and G.-Z. Yang, "Real-time activity classification using ambient and wearable sensors," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 6, pp. 1031–1039, 2009.

[57] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Tracking activities in complex settings using smart environment technologies," *International journal of biosciences, psychiatry, and technology (IJBSPT)*, vol. 1, no. 1, p. 25, 2009.

[58] E. Nazerfard and D. J. Cook, "Crafft: an activity prediction model based on bayesian networks," *Journal of ambient intelligence and humanized computing*, vol. 6, no. 2, pp. 193–205, 2015.

[59] D. Riboni, C. Bettini, G. Civitarese, Z. H. Janjua, and R. Helaoui, "Smartfaber: Recognizing fine-grained abnormal behaviors for early detection of mild cognitive impairment," *Artificial intelligence in medicine*, vol. 67, pp. 57–74, 2016.

[60] X. Hong, C. D. Nugent, M. D. Mulvenna, S. Martin, S. Devlin, and J. G. Wallace, "Dynamic similarity-based activity detection and recognition within smart homes," *International Journal of Pervasive Computing and Communications*, 2012.

[61] T. Van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *Proceedings of the 10th international conference on Ubiquitous computing*, 2008, pp. 1–9.

[62] D. Hao Hu, S. J. Pan, V. W. Zheng, N. N. Liu, and Q. Yang, "Real world activity recognition with multiple goals," in *Proceedings of the 10th international conference on Ubiquitous computing*, 2008, pp. 30–39.

[63] Y. Zhang, Y. Zhang, E. Swears, N. Larios, Z. Wang, and Q. Ji, "Modeling temporal interactions with interval temporal bayesian networks for complex activity recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 10, pp. 2468–2483, 2013.

[64] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, p. 832–843, Nov. 1983. [Online]. Available: https://doi.org/10.1145/182.358434

[65] J. Wen, S. W. Loke, J. Indulska, and M. Zhong, "Sensor-based activity recognition with dynamically added context," *EAI endorsed transactions on energy web*, vol. 15, no. 7, pp. 1–10, 2015.

[66] D. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," *Knowledge and information systems*, vol. 36, no. 3, pp. 537–556, 2013.

[67] S. W. Loke, "Representing and reasoning with situations for context-aware pervasive computing: a logic programming perspective," *The Knowledge Engineering Review*, vol. 19, no. 3, p. 213, 2004.

[68] D. Riboni and C. Bettini, "Owl 2 modeling and reasoning with complex human activities," *Pervasive and Mobile Computing*, vol. 7, no. 3, pp. 379–395, 2011.

[69] X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung, "Ontology based context modeling and reasoning using owl," in *IEEE annual conference on pervasive computing and communications workshops, 2004. Proceedings of the second*. Ieee, 2004, pp. 18–22.

[70] C. Pinhanez and A. Bobick, "Fast constraint propagation on specialized allen networks and its application to action recognition and control," *Submitted to AAAI*, vol. 98, 1998.

[71] V. Jakkula, D. J. Cook, and A. S. Crandall, "Temporal pattern discovery for anomaly detection in a smart home," 2007.

[72] C. Dousson and P. Le Maigat, "Chronicle recognition improvement using temporal focusing and hierarchization." in *IJCAI*, vol. 7, 2007, pp. 324–329.

[73] S. McKeever, J. Ye, L. Coyle, C. Bleakley, and S. Dobson, "Activity recognition using temporal evidence theory," *Journal of Ambient Intelligence and Smart Environments*, vol. 2, no. 3, pp. 253–269, 2010.

[74] J. Boger, P. Poupart, J. Hoey, C. Boutilier, G. Fernie, and A. Mihailidis, "A decision-theoretic approach to task assistance for persons with dementia," in *Ijcai*. Citeseer, 2005, pp. 1293–1299.

[75] F. Pecora, M. Cirillo, F. Dell'Osa, J. Ullberg, and A. Saffiotti, "A constraint-based approach for proactive, context-aware human support," *Journal of Ambient Intelligence and Smart Environments*, vol. 4, no. 4, pp. 347–367, 2012.

[76] S. Abburu, "A survey on ontology reasoners and comparison," *International Journal of Computer Applications*, vol. 57, no. 17, 2012.

[77] F. Baader, D. Calvanese, D. McGuinness, P. Patel-Schneider, D. Nardi *et al.*, *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003.

[78] D. L. McGuinness, F. Van Harmelen *et al.*, "Owl web ontology language overview," *W3C recommendation*, vol. 10, no. 10, p. 2004, 2004.

[79] G. Meditskos, S. Dasiopoulou, and I. Kompatsiaris, "Metaq: A knowledge-driven framework for context-aware activity recognition combining sparql and owl 2 activity patterns," *Pervasive and Mobile Computing*, vol. 25, pp. 104–124, 2016.

[80] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," *Journal of Web Semantics*, vol. 5, no. 2, pp. 51–53, 2007.

[81] F. Bobillo and U. Straccia, "The fuzzy ontology reasoner fuzzydl," *Knowledge-Based Systems*, vol. 95, pp. 12–34, 2016.

[82] R. Zese, *Probabilistic Semantic Web: Reasoning and Learning*. IOS Press, 2016, vol. 28.

[83] G. Meditskos, E. Kontopoulos, and I. Kompatsiaris, "Knowledge-driven activity recognition and segmentation using context connections," in *International Semantic Web Conference*. Springer, 2014, pp. 260–275.

[84] G. Okeyo, L. Chen, H. Wang, and R. Sterritt, "Dynamic sensor data segmentation for real-time knowledge-driven activity recognition," *Pervasive and Mobile Computing*, vol. 10, pp. 155–172, 2014.

[85] J. Ye and G. Stevenson, "Semantics-driven multi-user concurrent activity recognition," in *International Joint Conference on Ambient Intelligence*. Springer, 2013, pp. 204–219.

[86] D. Riboni and C. Bettini, "Cosar: hybrid reasoning for context-aware activity recognition," *Personal and Ubiquitous Computing*, vol. 15, no. 3, pp. 271–289, 2011.

[87] G. Meditskos, E. Kontopoulos, and I. Kompatsiaris, "Redef: Context-aware recognition of interleaved activities using owl 2 and defeasible reasoning." *Ssn-tc/ordring@ iswc*, vol. 1488, pp. 31–42, 2015.

[88] L. Ehrlinger and W. Wöß, "Towards a definition of knowledge graphs." *SEMANTiCS (Posters, Demos, SuCCESS)*, vol. 48, no. 1-4, p. 2, 2016.

[89] J. Euzenat, P. Shvaiko *et al.*, *Ontology matching*. Springer, 2007, vol. 18.

[90] M. Richardson and P. Domingos, "Markov logic networks," *Machine learning*, vol. 62, no. 1-2, pp. 107–136, 2006.

[91] D. Riboni, C. Bettini, G. Civitarese, Z. H. Janjua, and R. Helaoui, "Fine-grained recognition of abnormal behaviors for early detection of mild cognitive impairment," in *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2015, pp. 149–154.

[92] R. Helaoui, D. Riboni, and H. Stuckenschmidt, "A probabilistic ontological framework for the recognition of multilevel human activities," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, 2013, pp. 345–354.

[93] D. Riboni, T. Sztyler, G. Civitarese, and H. Stuckenschmidt, "Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016, pp. 1–12.

[94] A. Matassa and D. Riboni, "Reasoning with smart objects' affordance for personalized behavior monitoring in pervasive information systems," *Knowledge and Information Systems*, pp. 1–24, 2019.

[95] L. Chen, C. Nugent, and G. Okeyo, "An ontology-based hybrid approach to activity modeling for smart homes," *IEEE Transactions on human-machine systems*, vol. 44, no. 1, pp. 92–105, 2013.

[96] R. Mojarad, F. Attal, A. Chibani, S. R. Fiorini, and Y. Amirat, "Hybrid approach for human activity recognition by ubiquitous robots," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5660–5665.

[97] A. G. Salguero, P. Delatorre, J. Medina, M. Espinilla, and A. J. Tomeu, "Ontology-based framework for the automatic recognition of activities of daily living using class expression learning techniques," *Scientific Programming*, vol. 2019, 2019.

[98] A. G. Salguero and M. Espinilla, "Ontology-based feature generation to improve accuracy of activity recognition in smart environments," *Computers & Electrical Engineering*, vol. 68, pp. 1–13, 2018.

[99] K. Gayathri, K. Easwarakumar, and S. Elias, "Probabilistic ontology based activity recognition in smart homes using markov logic network," *Knowledge-Based Systems*, vol. 121, pp. 173–184, 2017.

[100] G. Civitarese, C. Bettini, T. Sztyler, D. Riboni, and H. Stuckenschmidt, "newnectar: Collaborative active learning for knowledge-based probabilistic activity recognition," *Pervasive and Mobile Computing*, vol. 56, pp. 88–105, 2019.

[101] J. Ye, G. Stevenson, and S. Dobson, "Usmart: An unsupervised semantic mining activity recognition technique," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 4, no. 4, pp. 1–27, 2014.

[102] F. De Backere, F. Ongenae, F. Van Den Abeele, J. Nelis, P. Bonte, E. Clement, M. Philpott, J. Hoebeke, S. Verstichel, A. Ackaert *et al.*, "Towards a social and context-aware multi-sensor fall detection and risk assessment platform," *Computers in Biology and Medicine*, vol. 64, pp. 307–320, 2015.

[103] A. Scalmato, A. Sgorbissa, and R. Zaccaria, "Describing and recognizing patterns of events in smart environments with description logic," *IEEE transactions on cybernetics*, vol. 43, no. 6, pp. 1882–1897, 2013.

[104] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing?" *International journal of human-computer studies*, vol. 43, no. 5-6, pp. 907–928, 1995.

[105] D. L. McGuinness and F. Van Harmelen, "OWL Web Ontology Language Overview," *W3C Recommendation*, vol. 10, 2004.

[106] F. Baader, I. Horrocks, C. Lutz, and U. Sattler, *Introduction to Description Logic*. Cambridge University Press, Apr. 2017.

[107] F. M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf, "Deduction in concept languages: From subsumption to instance checking," *Journal of logic and computation*, vol. 4, no. 4, pp. 423–452, 1994.

[108] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean *et al.*, "Swrl: A semantic web rule language combining owl and ruleml," *W3C Member submission*, vol. 21, no. 79, pp. 1–31, 2004.

[109] N. Abadie, A. Mechouche, and S. Mustière, "OWL-based formalisation of geographic databases specifications," vol. 674, oct 2010.

[110] E. Yuan, "Towards ontology-based software architecture representations," in *2017 IEEE/ACM 1st International Workshop on Establishing the Community-Wide Infrastructure for Architecture-Based Software Engineering (ECASE)*, May 2017, pp. 21–27.

[111] R. Stevens, M. E. Aranguren, K. Wolstencroft, U. Sattler, N. Drummond, M. Horridge, and A. Rector, "Using OWL to model biological knowledge," *International Journal of Human-Computer Studies*, vol. 65, no. 7, pp. 583–594, 2007.

[112] S. Lemaignan, M. Warnier, E. A. Sisbot, A. Clodic, and R. Alami, "Artificial cognition for social human–robot interaction: An implementation," *Artificial Intelligence*, vol. 247, pp. 45–69, Jun. 2017.

[113] A. Armand, D. Filliat, and J. Ibañez-Guzman, "Ontology-based context awareness for driving assistance systems," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, June 2014, pp. 227–233.

[114] M. Horridge and S. Bechhofer, "The OWL API: A Java API for OWL ontologies," *Semantic Web*, vol. 2, pp. 11–21, 2011.

[115] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson, "Jena: Implementing the semantic web recommendations," in *Proceedings of the 13th International World Wide Web Conference on Alternate track papers & posters*. New York, USA: ACM, May 2004, pp. 74–83.

[116] P. Eric and S. Andy, "SPARQL query language for RDF," World Wide Web Consortium (W3C), Tech. Rep., jan 2008, https://www.w3.org/TR/rdf-sparql-query/.

[117] I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," World Wide Web Consortium (W3C), Tech. Rep., May 2004, https://www.w3.org/Submission/SWRL/.

[118] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical OWL-DL reasoner," *Journal Web Semantics: science, services and agents on the World Wide Web*, vol. 5, no. 2, pp. 51–53, Jun. 2007.

[119] B. Glimm, I. Horrocks, B. Motik, G. Stoilos, and Z. Wang, "Hermit: An OWL 2 reasoner," *Journal of Automated Reasoning*, vol. 53, pp. 245–269, 2014.

[120] R. Zese, *Probabilistic Semantic Web: Reasoning and Learning*, ser. Studies on the Semantic Web. IOS Press, Dec. 2016, vol. 28.

[121] F. Bobillo and U. Straccia, "The fuzzy ontology reasoner fuzzyDL," *Knowledge-Based Systems*, vol. 95, pp. 12–34, Mar. 2016.

[122] V. Mascardi, V. Cordì, and R. Paolo, "A comparison of upper ontologies," in *Dagli Oggetti agli Agenti – Agenti e Industria: Applicazioni tecnologiche degli agenti software (WOA 2007)*. Genova, Italy: Seneca Edizioni, Torino, September 2007, pp. 55–64.

[123] M. Beetz, D. Beßler, A. Haidu, M. Pomarlan, A. K. Bozcuoğlu, and G. Bartels, "KnowRob 2.0 - A 2nd generation knowledge processing framework for cognition-enabled robotic agents," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, May 2018, pp. 512–519.

[124] M.-R. Tazari, F. Furfari, S. Hanke, O. Höftberger, D. D. Kehagias, M. Mosmondor, R. Wichert, and P. Wolf, "The universaal reference model for aal," vol. 11, pp. 610–625, 2012.

[125] R. Hoehndorf, L. Slater, P. N. Schofield, and G. V. Gkoutos, "Aber–OWL: a framework for ontology-based data access in biology," *BMC bioinformatics*, vol. 16, no. jen, p. 26, 2015.

[126] L. Buhmann, J. Lehmann, P. Westphal, and S. Bin, "DL-learner structured machine learning on semantic web data," in *Proceedings of The Web Conference 2018 (TheWebConf)*. Lyon, France: ACM Press, April 2018, pp. 467–471.

[127] H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen, "The protégé owl plugin: An open development environment for semantic web applications," in *Proceedings of the International Semantic Web Conference*. Hiroshima, Japan: Springer, nov 2004, pp. 229–243.

[128] A. Saghafi, "Visualizing ontologies – A literature survey," in *Graph-Based Representation and Reasoning: 22nd International Conference on Conceptual Structures*, ser. Lecture Notes in Computer Science, vol. 9717. Annecy, France: Springer, July 2016, pp. 204–221.

[129] H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner, "Ontology-based integration of information - A survey of existing approaches." in *Proceedings of Ontologies and Information Sharing at the International Joint Conferences on Artificial Intelligence (IJCAI)*, Seattle, Washington, USA, August 2001.

[130] H. Knublauch, D. Oberle, P. Tetlow, and E. Wallace, "A Semantic Web Primer for Object-Oriented Software Developers," W3C, W3C Working Group Note, 2006.

[131] S. S. Baset and K. Stoffel, "Object-oriented Software Modeling with Ontologies Around - A Survey of Existing Approaches," in *Proceedings of the 30th International Conference on Software Engineering and Knowledge Engineering (SEKE)*, San Francisco, California, USA, July 2018, pp. 29–45.

[132] G. Stevenson and S. Dobson, "Sapphire: Generating java runtime artefacts from owl ontologies," in *Proceedings of the 23rd International Conference on Advanced Information Systems Engineering (CAiSE)*. London, UK: Springer, June 2011, pp. 425–436.

[133] S. Baset and K. Stoffel, "Ontojit: Parsing native OWL DL into executable ontologies in an object oriented paradigm," in *Proceedings of the 13th Internaltional Workshop on OWL: Experiences and Directions (OWLED) and 5th International Workshop on Reasoner Evaluation (ORE)*. Bologna, Italy: Springer, November 2016, pp. 1–14.

[134] J.-B. Lamy, "Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies," *Artificial Intelligence in Medicine*, vol. 80, pp. 11–28, Jul. 2017.

[135] B. Motik, P. F. Patel-Schneider, B. Parsia, C. Bock, A. Fokoue, P. Haase, R. Hoekstra, I. Horrocks, A. Ruttenberg, U. Sattler *et al.*, "OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax," W3C recommendation, Tech. Rep., 2009.

[136] B. Motik, P. F. Patel–Schneider, and B. C. Grau, "OWL 2 Web Ontology Language Direct Semantics (Second Edition)," W3C recommendation, Tech. Rep., 2012.

[137] A. Kramer and K. Z. Kramer, "The potential impact of the covid-19 pandemic on occupational status, work from home, and occupational mobility," 2020.

[138] M. B. Omary, J. Eswaraka, S. D. Kimball, P. V. Moghe, R. A. Panettieri, K. W. Scotto *et al.*, "The covid-19 pandemic and research shutdown: staying safe and productive," *The Journal of clinical investigation*, vol. 130, no. 6, 2020.

[139] K. Gayathri, K. Easwarakumar, and S. Elias, "Probabilistic ontology based activity recognition in smart homes using Markov logic network," *in: Knowledge-Based Systems*, vol. 121, pp. 173–184, 2017.

[140] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Tracking activities in complex settings using smart environment technologies," *in: International Journal of Biosciences, Psychiatry, and Technology*, vol. 1, no. 1, p. 25, 2009.

[141] D. Dell'Aglio, E. Della Valle, F. van Harmelen, and A. Bernstein, "Stream reasoning: A survey and outlook," *Data Science*, vol. 1, no. 1-2, pp. 59–83, 2017.

[142] D. de Leng and F. Heintz, "Qualitative spatio-temporal stream reasoning with unobservable intertemporal spatial relations using landmarks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[143] A. Scalmato, A. Sgorbissa, and R. Zaccaria, "Describing and recognizing patterns of events in smart environments with description logic," *in: IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1882–1897, 2013.

[144] M. H. M. Noor, Z. Salcic, I. Kevin, and K. Wang, "Ontology-based sensor fusion activity recognition," *in: Journal of Ambient Intelligence and Humanized Computing*, pp. 1–15, 2018.

[145] R. Begg and R. Hassan, *Artificial Neural Networks in Smart Homes*. Berlin, Heidelberg: Springer, 2006, pp. 146–164.

[146] A. Fleury, M. Vacher, and N. Noury, "Svm-based multimodal classification of activities of daily living in health smart homes: sensors, algorithms, and first experimental results," *in: IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 2, pp. 274–283, 2010.

[147] E. Nazerfard and D. J. Cook, "CRAFFT: an activity prediction model based on Bayesian networks," *In: Journal of Ambient Intelligence and Humanized Computing*, vol. 6, no. 2, pp. 193–205, April 2015.

[148] D. Riboni, T. Sztyler, G. Civitarese, and H. Stuckenschmidt, "Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning," in *Proceedings of the 4th Association for Computing Machinery International Joint Conference on Pervasive and Ubiquitous Computing*. Heidelberg, Germany: ACM, September 2016, pp. 1–12.

[149] D. Hooda and R. Rani, "Ontology driven human activity recognition in heterogeneous sensor measurements," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 12, pp. 5947–5960, 2020.

[150] T. Kleinberger, M. Becker, E. Ras, A. Holzinger, and P. Müller, "Ambient intelligence in assisted living: enable elderly people to handle future interfaces," in *International conference on universal access in human-computer interaction*. Springer, 2007, pp. 103–112.

[151] G. Demiris, M. J. Rantz, M. A. Aud, K. D. Marek, H. W. Tyrer, M. Skubic, and A. A. Hussam, "Older adults' attitudes towards and perceptions of 'smart home'technologies: a pilot study," *Medical informatics and the Internet in medicine*, vol. 29, no. 2, pp. 87–94, 2004.

[152] M. Ziefle, C. Rocker, and A. Holzinger, "Medical technology in smart homes: exploring the user's perspective on privacy, intimacy and trust," in *2011 IEEE 35th Annual Computer Software and Applications Conference Workshops*. IEEE, 2011, pp. 410–415.

[153] M. Vacher, F. Portet, S. Rossato, F. Aman, C. Golanski, and R. Dugheanu, "Speech-based interaction in an aal context," 2012.

[154] M. Vacher, S. Caffiau, F. Portet, B. Meillon, C. Roux, E. Elias, B. Lecouteux, and P. Chahuara, "Evaluation of a context-aware voice interface for ambient assisted living: qualitative user study vs. quantitative system evaluation," *ACM Transactions on Accessible Computing (TACCESS)*, vol. 7, no. 2, pp. 1–36, 2015.

[155] F. Portet, M. Vacher, C. Golanski, C. Roux, and B. Meillon, "Design and evaluation of a smart home voice interface for the elderly: acceptability and objection aspects," *Personal and Ubiquitous Computing*, vol. 17, no. 1, pp. 127–144, 2013.

[156] A. Pradhan, K. Mehta, and L. Findlater, ""accessibility came by accident" use of voice-controlled intelligent personal assistants by people with disabilities," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1–13.

[157] V. Distler, C. Lallemand, and V. Koenig, "How acceptable is this? how user experience factors can broaden our understanding of the acceptance of privacy trade-offs," *Computers in Human Behavior*, vol. 106, p. 106227, 2020.

[158] A. P. Vermeeren, E. L.-C. Law, V. Roto, M. Obrist, J. Hoonhout, and K. Väänänen-Vainio-Mattila, "User experience evaluation methods: current state and development needs," in *Proceedings of the 6th Nordic conference on human-computer interaction: Extending boundaries*, 2010, pp. 521–530.

[159] S. Kujala, V. Roto, K. Väänänen-Vainio-Mattila, E. Karapanos, and A. Sinnelä, "Ux curve: A method for evaluating long-term user experience," *Interacting with computers*, vol. 23, no. 5, pp. 473–483, 2011.

[160] L. Arhippainen and M. Tähti, "Empirical evaluation of user experience in two adaptive mobile application prototypes," in *MUM 2003. Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia*, no. 011. Linköping University Electronic Press, 2003, pp. 27–34.

[161] E. L.-C. Law, "The measurability and predictability of user experience," in *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*, 2011, pp. 1–10.

[162] M. Turunen, J. Hakulinen, A. Melto, T. Heimonen, T. Laivo, and J. Hella, "Suxes-user experience evaluation method for spoken and multimodal interaction," in *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[163] E. L.-C. Law and P. Van Schaik, "Modelling user experience–an agenda for research and practice," *Interacting with computers*, vol. 22, no. 5, pp. 313–322, 2010.

[164] B. Laugwitz, T. Held, and M. Schrepp, "Construction and evaluation of a user experience questionnaire," in *Symposium of the Austrian HCI and Usability Engineering Group*. Springer, 2008, pp. 63–76.

[165] A. Hinderks, M. Schrepp, F. J. D. Mayo, M. J. Escalona, and J. Thomaschewski, "Developing a ux kpi based on the user experience questionnaire," *Computer Standards & Interfaces*, vol. 65, pp. 38–44, 2019.

[166] M. Schrepp, A. Hinderks, and J. Thomaschewski, "Construction of a benchmark for the user experience questionnaire (ueq)." *IJIMAI*, vol. 4, no. 4, pp. 40–44, 2017.

[167] ——, "Design and evaluation of a short version of the user experience questionnaire (ueq-s)." *IJIMAI*, vol. 4, no. 6, pp. 103–108, 2017.

[168] V. Jain and M. Singh, "Ontology development and query retrieval using protégé tool," *International Journal of Intelligent Systems and Applications*, vol. 9, pp. 67–75, 2013.

[169] S. E. Edgell and S. M. Noon, "Effect of violation of normality on the t test of the correlation coefficient." *Psychological bulletin*, vol. 95, no. 3, p. 576, 1984.

[170] M. Luck, P. McBurney, and C. Preist, "A manifesto for agent technology: Towards next generation computing," *Autonomous Agents and Multi-Agent Systems*, vol. 9, no. 3, pp. 203–252, 2004.

[171] A.-L. Barabási, "Network science," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 371, no. 1987, p. 20120375, 2013.

[172] S. Perera, M. G. Bell, and M. C. Bliemer, "Network science approach to modelling the topology and robustness of supply chain networks: a review and perspective," *Applied network science*, vol. 2, no. 1, pp. 1–25, 2017.

[173] S. Derrible and C. Kennedy, "The complexity and robustness of metro networks," *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 17, pp. 3678–3691, 2010.

[174] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: an open source software for exploring and manipulating networks," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 3, no. 1, 2009.

[175] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *science*, vol. 286, no. 5439, pp. 509–512, 1999.

[176] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[177] M. E. Newman, "A measure of betweenness centrality based on random walks," *Social networks*, vol. 27, no. 1, pp. 39–54, 2005.

[178] U. Brandes, "A faster algorithm for betweenness centrality," *Journal of mathematical sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[179] S. Fortunato and D. Hric, "Community detection in networks: A user guide," *Physics reports*, vol. 659, pp. 1–44, 2016.

[180] A. Lancichinetti and S. Fortunato, "Community detection algorithms: a comparative analysis," *Physical review E*, vol. 80, no. 5, p. 056117, 2009.

Appendix A

Representing modules of complex HAR system architectures as agents and analyzing them from the lens of network science

Complex HAR system architectures

Since a decade, there has been an active and rapid development in the domain of smart-homes. The services that a smart-home provides, can be divided into four main categories, namely, health-oriented [8], security-oriented [9], comfort/safety-oriented [10, 11], and energy-oriented [12]. Furthermore, to support health and well-being of older people and smart-home users in general, researchers and developers are actively working on companion robots [36] - see Section 5.4 in Chapter 5. In terms of the *services* that a smart-home provides, literature shows that some researchers focus on developing one particular service, whereas some others have focused on integrating and providing multiple services at the same time. Hence naturally, the software systems behind these smart-homes tend to be complex. The complexity increases as the number of services that a smart-home provides increases. Therefore, there are challenges that arise due to rising complexity of a software system. These challenges are most often architectural, i.e., they are challenges related to complexity, scalability, robustness, vulnerability (or fault-tolerance), etc.

Literature shows that, the above mentioned challenges are conventionally some of the challenges addressed by researchers in the domain of multi-agent systems (MAS) [170] using

the tools of network science [171]. For instance, in [172], the authors present a network science approach to modelling the topology and robustness of supply chain networks wherein agents are retailers, wholesalers, distributors and manufacturers. In [173], the authors evaluate the complexity and robustness of metro networks. If methodologies of network science are to be used for analyzing complex HAR systems then there is a need for certain remapping or in other words knowledge representation. We propose that the modules in a complex HAR system architecture can be represented as agents. For this, a knowledge-based approach can be taken, and an ontology can be developed to describe the domain. For instance, the terms *agent*, *system*, and the possible *interactions between the agents* can be described in the T-box and R-box of the ontology. Instead, the A-box could contain data related to the system's modules as agents. Furthermore, if some sort of reasoning is also to be taken into account then for computational benefit a multi ontology network could also be designed.

Representing modules of a system's architecture as agents

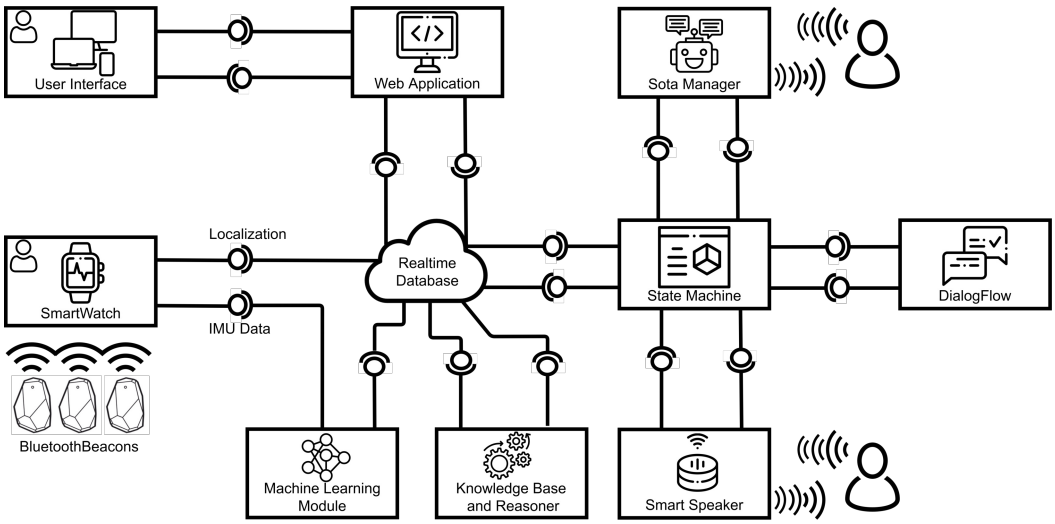


Figure A.1 An example of a complex HAR system's architecture.

In the previous section, we propose that if methodologies of network science are to be used for analyzing complex HAR systems then there is a need for certain remapping or in other words knowledge representation. In such a representation, modules of a complex HAR system can be represented as agents in a multi-agent system. Hence, in Figure A.1, we present an example of a complex HAR system architecture. It is an architecture that we have developed as part of our previous work, which is described in Section 5.4 of Chapter 5. In this system, HAR provides contextual information to digital assistants so that they can

proactively and contextually support the smart-home users. The digital assistants in this example are of two kinds, one is a robot-assistant and the other is a vocal assistant.

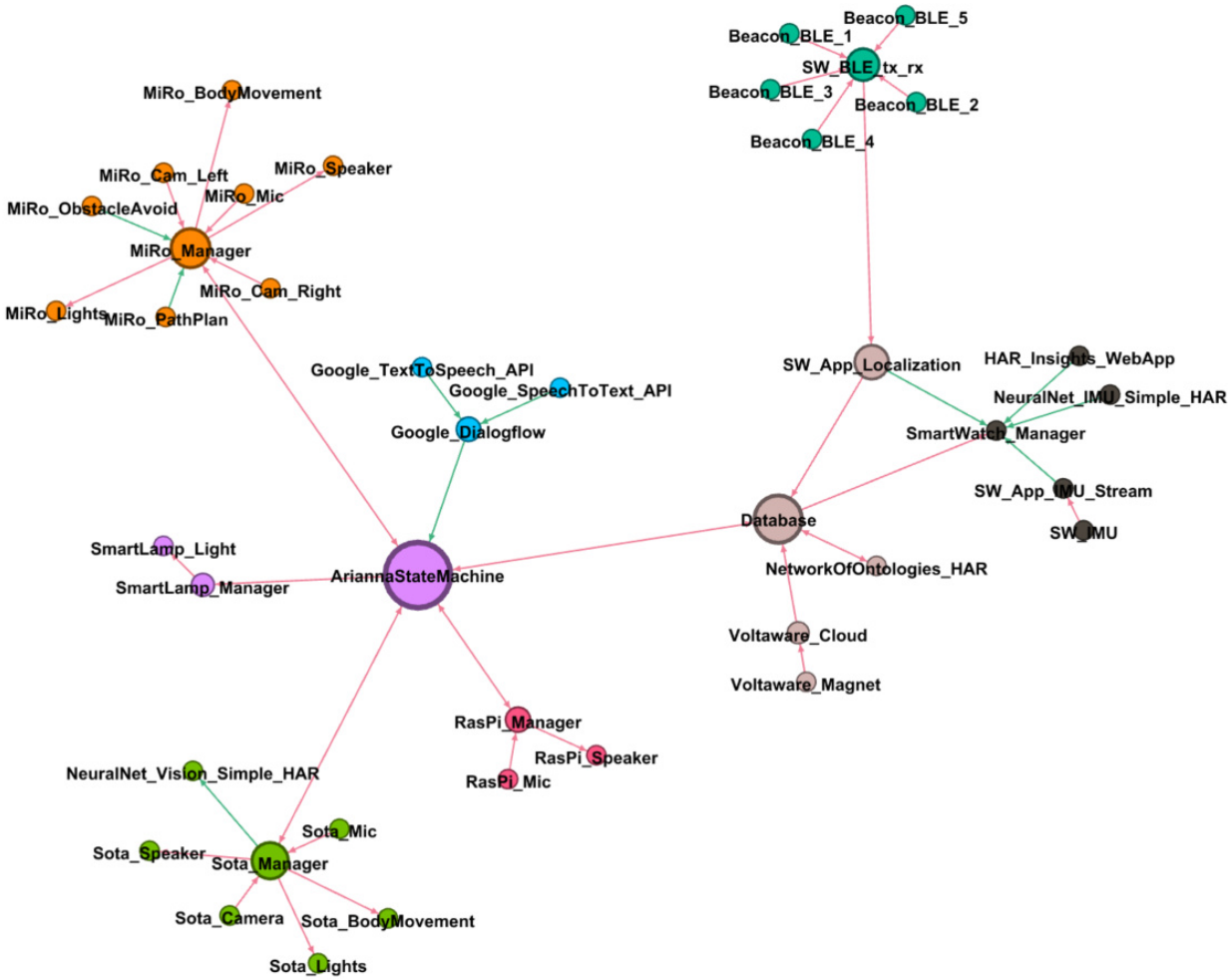


Figure A.2 A network developed using Gephi, where nodes represent agents and links represent communication between the agents. The color of nodes represents different communities and the color of links represents the type of communication between them.

Consider that the modules of the complex HAR system architecture presented in Figure A.1 are broken down into further smaller modules and represented as agents connected to each other as can be seen in the network presented in the Figure A.2. The network is designed using Gephi [174], which is an open-source network analysis and visualization software tool. In the network, the nodes represent agents (i.e., small modules which are part of a complex HAR system architecture) and links represents communication between the agents. On the one hand, for the nodes, the color represents different communities. The color is generated by running a community detection algorithm in Gephi and the size of a

node is based on the betweenness-centrality measure calculated by running an algorithm in Gephi. On the other hand, for the links, the color represents the type of communication between them. The color is defined by the knowledge engineer. The pink-color link represents a publish-subscribe form of communication and the green-color link represents a request-response form of communication.

Analyzing a multi-agent system from the lens of network science

Our preliminary research reveals that methodologies of network science can enable exploratory analysis of large and complex HAR systems, given that the modules of the system's architecture are represented as agents. Thus the representation makes the HAR system a multi-agent system. As is done in the literature, a multi-agent system can then be analyzed using methodologies of network science on three different *levels of granularity*.

Firstly, an analysis on the *microscopic properties* (e.g., in-degree, out-degree) of the network can give insights into the multi-agent system's complexity, robustness. Secondly, an analysis on the *macroscopic properties* (e.g., scale-free [175], small-world phenomena [176]) of the network can give insights into the multi-agent system's robustness to error propogation, i.e., fault-tolerance. Thirdly, an analysis on the *mesoscopic properties* (e.g., betweenness-centrality, community detection) of the network can give insights into the multi-agent system's abstraction, i.e., modularity.

Formally, lets consider the network of agents communicating with each other (shown in Figure A.2), as a *directed graph* $G(N, L)$, where N represents the *total number of nodes* and L represents the *total number of links* between the nodes. Nodes represent agents and links represent communication between agents. Note that in scientific literature the terms network and graph are used interchangeably [171]. The following equations enable us to analyze the properties of the network in three layers of granularity, i.e. microscopic, macroscopic and mesoscopic.

Microscopic level – at this level we analyse the properties of the nodes. A key property of each node is its degree, which represents the number of links it has with other nodes. For directed graphs, there is a difference between *incoming degree* k_i^{in}, which represents the number of links that point to node i, and *outgoing degree* k_i^{out}, which represents the number of links that point from node i to other nodes. Finally, for a node i, its *total degree* k_i^{tot} is given by

$$k_i^{tot} = k_i^{in} + k_i^{out}. \tag{A.1}$$

The total number of links in a directed graph G is

$$L = \sum_{i=1}^{N} k_i^{in} = \sum_{i=1}^{N} k_i^{out}. \tag{A.2}$$

The *average degree* of a directed graph G is

$$\left\langle k^{in} \right\rangle = \frac{1}{N} \sum_{i=1}^{N} k_i^{in} = \left\langle k^{out} \right\rangle = \frac{1}{N} \sum_{i=1}^{N} k_i^{out} = \frac{L}{N}. \tag{A.3}$$

An *adjacency matrix* is a mathematical representation of a network G, which allows to keep track of all the links L between all the nodes N within the network. Considering that we have a directed network G, the adjacency matrix of G (with N nodes) has N rows and N columns. The adjacency matrix A_{ij} has its elements as $A_{ij} = 1$, if there is a link pointing from node j to node i, and $A_{ij} = 0$, if there is no link between nodes i and j. Furthermore, for a directed network G without self-loops (i.e., a node having a link pointing to itself), $A_{ij} \neq A_{ji}$ and $A_{ii} = 0$.

Based on the adjacency matrix of a directed network G, for a node i, the number of incoming and outgoing degrees is the sum over the adjacency matrix's rows and columns, respectively

$$k_i^{in} = \sum_{j=1}^{N} A_{ij}, \qquad k_i^{out} = \sum_{j=1}^{N} A_{ji}. \tag{A.4}$$

The total number of links L in directed graph G is

$$L = \sum_{i,j=1}^{N} A_{ij}. \tag{A.5}$$

Macroscopic level – at this level we analyze properties of the whole network. A property as simple as the *degree distribution* can give us a glimpse into the structure of a network and allow us to distinguish different types of networks. In a directed network G, nodes can have a value for incoming degree k^{in} and a value for outgoing degree k^{out}. Therefore, the degree distribution becomes a two-dimensional probability distribution

$$P_{deg}(k^{in}, k^{out}) = \frac{N_{k^{in},k^{out}}}{N} \tag{A.6}$$

where, $N_{k^{in},k^{out}}$ represents the number of nodes with in-degree k^{in} and out-degree k^{out}. A two-dimensional histogram can be used to visualize a two-dimensional degree distribution, e.g., a surface plot or a color plot.

Apart from considering the correlation between the in-degree and out-degree using the degree distribution, we can also consider marginal degree distributions. One of the marginal degree distributions is the in-degree distribution

$$P^{in}_{deg}(k^{in}) = \frac{N_{k^{in}}}{N} \tag{A.7}$$

where, $N_{k^{in}}$ represents the number of nodes with in-degree k^{in}. The other is the out-degree distribution

$$P^{out}_{deg}(k^{out}) = \frac{N_{k^{out}}}{N} \tag{A.8}$$

where, $N_{k^{out}}$ represents the number of nodes with out-degree k^{out}. Another one-dimensional distribution that could be useful to analyse is the total degree distribution

$$P^{tot}_{deg}(k^{tot}) = \frac{N_{k^{tot}}}{N} \tag{A.9}$$

where, $N_{k^{tot}}$ represents the number of nodes with total-degree k^{out}.

Mesoscopic level – at this level we analyse properties that exist in between the microscopic level and macroscopic level of the network. One could consider, the measure of *betweenness-centrality* [177, 178] and *communities* [179, 180] in the network as properties on a mesoscopic level of the network.