ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Dottorato di Ricerca in
COMPUTER SCIENCE AND ENGINEERING
XXXV Ciclo

Settore Concorsuale: 01/B1 – INFORMATICA
Settore Scientifico Disciplinare: INF/01

INTEGRATION OF HETEROGENEOUS DATA SOURCES AND AUTOMATED REASONING IN HEALTHCARE AND DOMOTIC IOT SYSTEMS

Presentata da:
ROBERTO REDA

Coordinatore Dottorato:
ILARIA BARTOLINI

Supervisore:
ANTONELLA CARBONARO

Co-supervisore:
VITTORIO MANIEZZO

ESAME FINALE - ANNO 2023

KEYWORDS

Semantic Web

Internet of Things

Cognitive Computing

Healthcare

Domotics

Acknowledgements

First and foremost, I want to express my deepest gratitude to my supervisor, professor Antonella Carbonaro, for introducing me to the fascinating field of Semantic Web, for her continuous guidance, motivation and trust.

I am extremely grateful to professor Victor De Boer for his support and appreciation shown in my research, and for the long and fruitful collaboration during the last part of my PhD programme. My gratitude extends also to the other members of the Interconnect team (VU & TNO), Ronald Siebes, Roderick van der Weerdt, Barry Nouwt and Laura Daniele for all the insightful discussions we had. I also take the chance to thank professor Frank van Harmelen for reviewing this thesis.

I would like to thank my colleague and coauthor professor Filippo Piccinini for his guidance and advice during my early years as a scientific researcher. My sincere gratitude goes also to all the authors of research articles who inspired me with their works.

Finally, a special thank goes to my mum for her inestimable encouragement and support throughout my academic career.

Abstract

In recent years, IoT technology has radically transformed many crucial industrial and service sectors such as healthcare. The multi-facets heterogeneity of the devices and the collected information provides important opportunities to develop innovative systems and services. However, the ubiquitous presence of data silos and the poor semantic interoperability in the IoT landscape constitute a significant obstacle in the pursuit of this goal. Moreover, achieving actionable knowledge from the collected data requires IoT information sources to be analysed using appropriate artificial intelligence techniques such as automated reasoning. In this thesis work, Semantic Web technologies have been investigated as an approach to address both the data integration and reasoning aspect in modern IoT systems. In particular, the contributions presented in this thesis are the following: (1) the IoT Fitness Ontology, an OWL ontology that has been developed in order to overcome the issue of data silos and enable semantic interoperability in the IoT fitness domain; (2) a Linked Open Data web portal for collecting and sharing IoT health datasets with the research community; (3) a novel methodology for embedding knowledge in rule-defined IoT smart home scenarios; and (4) a knowledge-based IoT home automation system that supports a seamless integration of heterogeneous devices and data sources.

Contents

Introduction

Motivation and Contribution

During the last decades the Internet of Things (IoT) technology has radically transformed many industrial and service sectors including manufacturing, transportation, energy management and home automation [150]. Among all of them, an important and crucial area for IoT applications is the healthcare field [191]. Indeed, since the early stages of IoT technology development the potential of IoT devices in the healthcare sector has always attracted a lot of interest from both the industry and the academia [26, 90].

Notably, the fitness industry has assisted to an unprecedented proliferation on the market of consumer IoT devices such as fitness trackers and smartwatches. Wearable devices, initially intended for keeping track of training sessions, are nowadays employed to constantly monitor a lot physiological parameters of the wearers including the heart-beat, sleep cycles and daily physical activities [122].

From a data-centric point of view, IoT fitness and wellness devices constantly collect and store on the cloud an enormous amount of users' personal health data. All of this information constitutes an invaluable resource for researches and domain experts because, if properly analysed, it can provide better insights into our health. Moreover, the integration of IoT wearables devices with other emerging IoT technologies such as home automation can potentially lead to the development of innovative and more efficient crucial applications such as smart assisted living technologies [36].

However, due to the lack of common adopted standards and communication protocols (which results in poor interoperability and data integration issues), the potential of IoT devices is still far from being fully exploited. For example, the inevitable ubiquitous presence of data silos in the IoT healthcare landscape prevents users, health professionals and researchers from getting an essential integrated view of the collected health and fitness data [143, 232]. Moreover, existing IoT applications are still highly dependent on human beings for the cognition processing (i.e., the decision making and taking actions process) whereas cognitive computing techniques could significantly enhance data analysis in order to achieve actionable knowledge from data [226, 222].

A promising approach for data integration and reasoning in IoT systems comes from the Semantic Web (SW) technologies [31]. SW technologies consist of a set of recommended languages and best practises for describing data and formally representing domain knowledge. A considerable amount of research has shown that SW technologies constitute an appropriate means for achieving data interoperability in heterogeneous systems including IoT [24, 133]. Moreover, SW technologies natively enable automated reasoning capabilities over the integrated semantic-enriched data, thus allowing a higher level of abstraction that could not be otherwise obtained using other traditional programming paradigms.

The objective of this thesis work was to study a SW technologies based approach to tackle the data heterogeneity issues in IoT fitness devices and investigate innovative and efficient methodologies for reasoning over integrated disjoint-domains IoT data [226].

The resulting contribution of this thesis work is twofold. The former addresses the issues of data silos and data interoperability in the IoT fitness domain. The IoT Fitness Ontology (IFO) has been designed in order to integrate and homogenise the heterogeneity of health data collected by IoT fitness devices. Specifically, the IFO ontology formally describes the most common and important concepts in the domain and the relationships among them. Moreover, the IFO ontology was employed to develop a Linked Open Data (LOD) web portal for collecting and sharing IoT health datasets with the research community.

The latter regards the use of SW technologies data analysis of disjoint-domains integrated data sources in IoT-based home automation systems. In particular, a novel approach of using the Semantic Web Rule Language (SWRL) as a smart home scenario programming paradigm has been devised. The idea behind the proposed methodology is to exploit the reasoning capabilities offered by the SW technologies to overcome the limitations of the predominant trigger-action model that hamper the full exploitation of IoT devices in home automation systems. The Semantic Smart Home System (SSHS) was developed in order to support the execution of the knowledge enhanced rule-based scenarios. Experimental results have shown the feasibility and the efficiency of the proposed approach in real-life settings.

Thesis Organisation

This thesis is mainly divided in two parts. The first one provides background information on the SW technologies and the IoT systems that are the main object of this study. The second one regards the contributions of the study. A brief description of each chapter is provided in the following paragraphs.

Part I: Background

Chapter 1 presents a detailed overview of the current state of art of Semantic Web. The Semantic Web architecture is analysed by breaking it into its component parts. It introduces the main concept of ontologies and the role they play within the context of the Semantic Web. The Web Ontology Language (OWL) and the Semantic Web Rule Language are described in depth. Moreover, it draws attention to the concept of Open Data and its collocation in the context of the Semantic Web.

Chapter 2 provides an introduction to the Internet of Things technologies, in particular the role of the Internet of Things in the healthcare and fitness domain. Critical aspects of IoTs such as interoperability issues, from a data-centric perspective, are taken into a detailed consideration. Secondly it offers an overview of the most common IoT fitness and smart home devices available on the market.

Part II: Contribution

Chapter 3 introduces the problem of data silos and data interoperability issues in the IoT fitness landscape which are the main motivation of this study. It illustrates in detail the structure and the peculiar characteristics of the IFO ontology. Furthermore, it presents the LOD portal that was developed for collecting and sharing IoT health datasets.

Chapter 4 highlights the limitations of the predominant trigger-action model that hamper the full exploitation of IoT devices in home automation systems. It illustrates the novel methodology devised for defining smart home scenarios using SWRL and the supporting semantic smart home system for their execution. Moreover, it describes the numerous experiments that were carried out to prove the feasibility and the efficiency of the proposed approach in realistic settings.

Chapter 5 summarises the main contributions of this thesis study and outlooks several possible future research directions.

List of Publications

The findings and results of the work presented in the contribution part of this thesis have been disseminated in the following publications.

Journal Articles

- Reda, R., Piccinini, F., Martinelli, G., & Carbonaro, A. (2022). Heterogeneous self-tracked health and fitness data integration and sharing according to a linked open data approach. *Computing*, 104(4), 835-857. [Impact factor: 2.220 (2020)]

- Reda, R., Carbonaro, A., de Boer, V., Siebes, R., van der Weerdt, R., Nouwt, B., & Daniele, L. (2022). Supporting Smart Home Scenarios Using OWL and SWRL Rules. *Sensors*, 22(11), 4131. [Impact Factor: 3.576 (2020)]

Part I

Background

Chapter 1

Semantic Web Technologies

This chapter provides an overview of the current state of art of Semantic Web technologies. The Semantic Web architecture is analysed by breaking it into its component parts. In particular, RDF, OWL, SWRL and RML languages are described in depth. Moreover, this chapter draws the attention to the concepts of ontology and automated reasoning, and their collocation in the context of the Semantic Web.

1.1 The Semantic Web

The World Wide Web (simply known as Web) has been developed back to 1990 by Tim Berners-Lee at CERN in Geneva, Switzerland [41].

The innovative idea behind the Berners-Lee's seminal work was to use the hypertext technology [184] as a means to realise a distributed global system of interlinked documents accessible via the Internet.

On the Web, documents and resources are univocally identified by Uniform Resource Locator (URL) addresses which specify how they can be retrieved across the Internet from their remote location. Documents are interconnected to each other by means of hyperlinks and URLs of the target resources are directly embedded in the body text.

The HyperText Markup Language (HTML) is used to define the structure of the documents which primarily contain information in natural language, digital images, multimedia resources along with the rendering instructions to be displayed for human consumption.

Since its appearance on the Internet, the World Wide Web has become more and more mainstream and has grown into the world's largest repository of human knowledge. The rapid growth of the amount of information on World Wide Web has raised many research challenges such as information overloading, poor retrieval and aggregation problems. To find useful information is like trying *"to find a needle in a haystack"* for humans, due to the huge amount of data available and a hard task even for search engines which rely mostly on content-independent statistical algorithms. Syntactic variations or misspellings of the search keywords in documents prevent a reliable statistical score of document relevance.

Furthermore, users are often interested to retrieve data in aggregated manner instead of single separated documents. For instance, a user might be interested to find a smartwatch with certain features at the lowest price on the market. Performing such a task requires to gather information form several companies web pages, integrating their content and a kind of reasoning about the data obtained.

These issues derive from the fact that the current Web is mainly designed for human consumption and not for an automated machine processing, that is web pages do not provide any semantic information about the content which could allow machines to determine what the page content means.

The *Semantic Web* is an emerging research area which aims to overcome the challenge of allowing humans and computers to cooperate in the same way humans cooperate with each other.

Tim Berners-Lee, the Web's inventor, has coined the term Semantic Web and in [43] provides a concise definition of it: *"The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation"*.

Berners-Lee envisages the World Wide Web as a collaborative medium by which users can share information and services easily and aggregating data from different sources where documents and web pages are understandable and processable by machines.

Even though the original vision of the Semantic Web is still far from being completely realised, Semantic Web technologies have matured over the years, and there are nowadays available a number of solutions and tools to efficiently deal the semantic data. Moreover, Semantic Web technologies have also been applied to technologies and systems, which were not originally meant for, to provide interoperable interface, process, and service descriptions [31].

1.1.1 The Semantic Web Architecture

The Semantic Web Architecture, as shown in Figure 1.1, is based on a layered approach, and each layer provides a set of specific functionalities. Several standards and technologies contribute to the realisation of the Semantic Web.

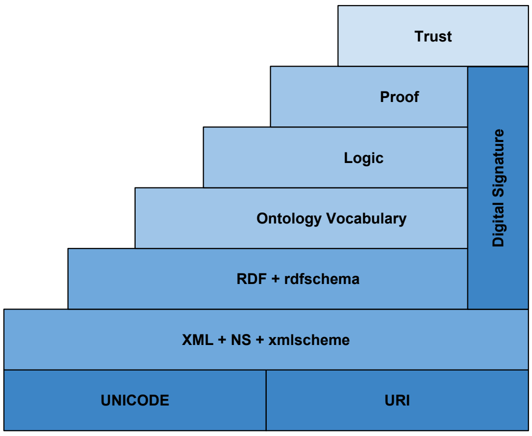

Figure 1.1: Semantic Web Architecture. (Image adapted from [221]).

The lowest layers consist of data and metadata and provide a standard representation for information so that data can be easily exchanged among heterogeneous systems and applications.

The UNICODE provides a standard for a consistent encoding and representation of text expressed in most of the world's writing systems [71].

The URI provides a simple and extensible means for identifying and locating remote resources, such as web pages, media contents or other forms of data on the World Wide Web [167].

XML, the standard syntax for representing information in the Web, allows to structure data by means of user-defined tags and data interoperability [55].

The Resource Description Framework (abbreviated RDF) describes the information contained in a Web resource providing unambiguous methods to express semantics [174].

RDF Schema (abbreviated RDFS) allows to define simple vocabularies used in RDF descriptions [57].

Semantic layers, on the top of the stack, include ontology languages, rule languages, query languages, logic, reasoning mechanisms, and trust.

Ontologies constitute the backbone of the Semantic Web. Ontologies are a means to express concepts of a given domain and the relationships among the concepts and they also specify complex constraints on the types of resources and their properties.

OWL, the most popular ontology language, is an extension of RDFS. OWL Lite, OWL DL, and OWL Full are the three sub-languages of the OWL family ontology [171].

Rule languages allow writing inference rules in a standard way which can be used for reasoning in a particular domain. Among several standards of rule languages there are RuleML and SWRL (Semantic Web Rule Language) [130]. The latter combines RuleML and OWL, and includes a high-level abstract syntax for Horn-like rules.

SPARQL, a standardised query language for RDF data, provides both a protocol and a language for querying RDF graphs via pattern matching [200].

On the highest layers there are logic and reasoning, logic provides the theoretical underpinning required for reasoning and deduction. First order logic and description logic are frequently used to support the reasoning system which can make inferences and extract new insights based on the resource content rely on one or more ontologies.

Trust, Security, are needed to assure that the information content of resources is of high quality and can be trusted. More research is still to be done in order to develop comprehensive solutions and techniques to assess and ensure the trustworthiness, security, and privacy of Semantic Web content.

1.2 Resources Description

In the Semantic Web, resources can be either abstract concepts such as ideas or thoughts, or concrete objects such as people, devices or images. Technically, a resource is anything can be univocally identified with a Universal Resource Identifier (URI). For example, the Friend of a Friend (FOAF) vocabulary uses *http://xmlns.com/foaf/0.1/mbox* to represent the concept of a person's email address. Once resources are univocally identified, they can be retrieved, linked together and semantically described.

1.2.1 Resource Description Framework

The Resource Description Framework (RDF) is a language for describing metadata about the resources and a W3C recommendation [163].

Given that a resource is anything that can be referenced by a URI [42], RDF is suitable to describe a resource of any type even when the resource can not be directly accessed from the Web [163].

RDF is mainly intended to be used when data need to be machine processable rather than being only accessed by people. Furthermore, RDF provides standardised way to express information such that it can be exchanged between different systems without loss of meaning [163].

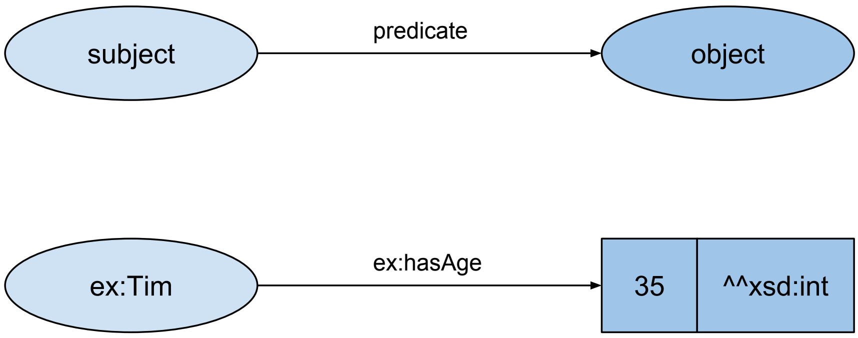

Figure 1.2: RDF graph of a generic triple and an example.

RDF describes resources by means of triples. RDF triples have the form `(subject, predicate, object)` and provide the way to make statements about things.

Statements define the properties of the resources. A property expresses a relationship between the subject and the object. A property can designate a class to a resource, define a literal value attribute of a resource and a relationship between two resources.

The following example shows an RDF triple:

```
ex:Tim ex:hasAge "35"^^xsd:int .
```

An RDF graph of the example above is depicted in Figure 1.2 along with a generic RDF triple.

Resources can be named or unnamed, the latter are represented with *blank nodes*.

```
_:bnode1 rdfs:label "anonymous" .
```

In the example above, `_:bnode1` denotes an anonymous resource, the prefix _ is used in many different RDF serialisation syntaxes to specify a blank node.

Naming and consistency are a significant part of RDF, user-defined resources are named using URIs and RDF supports CURIE syntax, which is an abbreviated syntax for expressing URIs [45]. For instance, given a prefix `ex:`, which acts as a shortcut for the URI `http://example.com/ontology#`, then `ex:heartRate` can be used instead of `http://example.com/ontology#heartRate`.

Datatypes in RDF are inherited from the existing XML Schema standard which defines a hierarchy of datatypes along with their syntax [46].

Language tagged strings in RDF should be defined in accordance with RFC 3066 [22] as shown in the following example:

```
ex:Italy
   rdfs:label "Italy"@en ;
   rdfs:label "Italia"@it .
```

RDF defines a core set of terms for describing resources, one of the most relevant is `rdf:type` which is used to state that a resource is a member of a specified class.

```
ex:Tim rdf:type foaf:Person .
```

The statement above asserts that the resource `Tim` is a member of the class `foaf:Person`.

RDF allows also to define containers which are used to describe groups (ordered, unordered or alternatives) of things with informally defined semantics [163], however since they are not widely used in practice they have been suggested as candidates for deprecation [96].

RDF collections are used to describe group that contains only the specified members. Unlike containers, collections may be closed and this is an important characteristic for reasoning.

Reification in RDF (describing RDF statements using RDF itself) is possible using the built-in terms: `rdf:Statement`, `rdf:subject`, `rdf:predicat` `e` and `rdf:object`.

Applications may need to describe RDF statements, for instance, to record information like when statements were made, or who made them; other use cases where reification is useful are discussed in [157].

```
ex:TimAgeTriple rdf:type rdf:Statement .
ex:TimAgeTriple rdf:subject ex:Tim .
ex:TimAgeTriple rdf:predicate ex:hasAge .
ex:TimAgeTriple rdf:object "35"^^xsd:int .
ex:TimAgeTriple ex:expires "2017-12-15"^^xsd:date .
```

The example above shows the reification of the statement:

```
ex:Tim ex:hasAge "35"^^xsd:int .
```

Even though properties in RDF are only binary relations (i.e., relations between two classes), n-ary relations, to link an individual to more than just one individual or value are possible by creating an intermediate entity that serves as the subject for the entire set of relations [185].

The following example introduces a blank node to model a tertiary relationship:

```
ex:Tim ex:hasHeight _:bnode1 .
_:bnode1 rdf:value ex:Measure .
_:bnode1 ex:numericalValue "186"^^xsd:int .
_:bnode1 ex:Unit ex:cmUnit .
```

RDF triples can also be put together to form larger networks also known as semantic networks. A semantic network is a direct graph where vertex are the subject or the object of a triple and edges are labelled with predicates and are directed from the subject to the object.

RDF is an abstract model and RDF statements can be represented either as a graph or in a textual format also called RDF serialisations. The most important RDF encoding syntax is RDF/XML [34] which is based on Extensible Markup Language (abbreviated XML) standard [55] and currently is the only normative RDF encoding standard. Other notable RDF serialisations syntax are: N-Triples [61] which is a line-based (a single statement cannot span multiple lines) plain text serialisation, Turtle and RDFa which allow to embed RDF statement within an XHTML document [15].

Terse RDF Triple Language

A notable RDF textual format representation, besides the most common serialisation RDF/XML, is Terse RDF Triple Language, abbreviated Turtle [33].

Turtle defines a syntax which allows a completely compact textual representation of RDF graphs, in both machine and human readable format.

Turtle syntax has also been used extensively throughout this thesis.

The salient characteristics of the Turtle syntax are briefly review:

- URIs are written surrounded by `< >` brackets.

  ```
  <http://example.com/fitnessOntology#Walking>
  ```

  This statement represents a walking activity entity.

- Namespaces can be declared to prefix URI using `@prefix`

  ```
  @prefix fo: <http://example.com/fitnessOntology#> .
  ```

- Tokens and terms are white-space delimited and triples are delimited by a `.` period character.
- Literals are represented between `" "` double-quotes.
- Literals can be typed by XSD datatypes; assigned datatypes are appended after a `^^` operator.
- The _ underscore prefix is used to denote blank nodes.

```
_:bnode1 rdfs:comment "anonymous"^^xsd:string .
```

- The term `a` can be used as a shortcut for `rdf:type`.

```
ex:Tim a foaf:Person .
```

- Triples which share a common subject and predicate can be grouped together using a `,` comma delimiter.
- Square brackets `[]` can alternately be used to denote blank nodes.

```
[ rdfs:comment "blank node"^^xsd:string,
               "another comment"^^xsd:string ] .
```

- Triples which share a given subject can be grouped together using a `;` semi-colon delimiter.

```
ex:Tim a foaf:Person;
        rdfs:comment "someone"^^xsd:string .
```

1.2.2 Resource Description Framework Schema

The Resource Description Framework Schema (RDFS or RDF Schema) [57] is a language for defining simple *vocabularies* (which are a kind of ontology) of terms that can be use to construct RDF statements according to these ontologies.

RDF Schema is an extension of RDF, it is expressed in RDF syntax, and provides the means for specifying well defined relationships between classes and properties in a hierarchical structure.

RDFS allows users to define classes and properties (predicates) using the relations `rdfs:Class` and `rdfs:Property`. A *class* is a set of things, sharing common characteristics, that we want to represents; a property is a binary relation between two class individuals. Individuals are instances of a class, which means that they are objects that belong to a particular class, are defined by assigning the type of a class to the resource through `rdf:type`.

In RDFS a class `C` is defined by a triple of the form:

```
C rdf:type rdfs:Class .
```

For example a class to represent "users" can be as follows:

```
ex:User rdf:type rdfs:Class .
```

A class in RDFS represents a set of resources and the hierarchy defines the relationship between different classes. RDFS is structured around the notion of a class hierarchy. A subclass is a class that has to be intended as a subset of the more general class and is specified by the property `rdfs:subClassOf`. The subclass relation is also the only relationship between classes that RDFS allows.

```
ex:User rdfs:subClassOf ex:Person .
```

This states that any member (also called instance) of the class `ex:User` is also a member of the class `ex:Person`.

In a similar way, RDFS allows the definition of a hierarchical structure also for properties in addition to the hierarchy of classes. That is, using the relation `rdfs:subPropertyOf` we can state that a property is more specialised than another.

```
ex:directorOf, rdfs:subPropertyOf, ex:worksFor .
```

This triple states that two objects related by the `ex:directorOf` property are also related by the `ex:worksFor` property.

Furthermore, RDFS allows to put restrictions on the properties to a certain classes of resource using the relations `rdfs:domain` and `rdfs:range`; which means that the domain and range of the property is restricted to specific classes.

Other properties introduced to make RDFS document more human-readable are: `rdfs:comment` which allows to give an informal description of the resource, `rdfs:label` for specifying an alternative labelling scheme, `rdfs:seeAlso` to reference another resource which provides related information and `rdfs:isDefinedBy` which is also a subproperty of the former and used to indicate that the definition of the resource is given elsewhere (e.g., in a book).

It is worth to mention that RDFS schema definitions are not prescriptive [192]. The RDFS schema is a merely description of the structure of the knowledge and it is let to the external application to decide whether to insist on full compliance with the schema or not. Because of the flexible nature of Semantic Web knowledge, it is perfectly acceptable to structure the knowledge base adding classes or properties outwith the schema or even violate specific constrains.

1.2.3 Linked Open Data

The idea behind the Open Data (abbreviated OD) is closely similar to the concept of the open source software [62]. According to the Open Definition the essence of open data can be summed up in the statement: *"Open means anyone can freely access, use, modify, and share for any purpose (subject, at most, to requirements that preserve provenance and openness)"* [5].

Jansenn et al. define Open Data as *"non-privacy restricted and non-confidential data which is produced with public money and is made available without any restrictions on its usage or distribution. Data can be provided by public and private organisations, as the essence is that the data is funded by public money"* [137].

Open Data refers to publish any collection of data in a machine-readable format, with no licensing or patent restriction so that everyone is free to use, reuse and redistribute for any purpose.

Governmental organisations, individuals, companies and enterprises are continuously gaining interest in Open Data recently. Governments provide

Table 1.1: 5-star Open Data

Stars	Data Characteristics
★	open license, any format
★★	structured format
★★★	non-proprietary open format
★★★★	URIs to identify resources
★★★★★	data interlinked to provide context

transparency and increase increase public participation through Open Data. Scientific institutions can benefit from Open Data for deriving new knowledge and insights. Entrepreneurs can use the data to support their business, strategic decisions and foster innovations.

An exhaustive survey about Open Data benefits and the challenges in adoption of it can be found in [137].

Strictly related to the concept of Open Data is the concept of *Linked Data* (abbreviated LD). Linked Data refers to *"data published on the Web in such a way that is machine-readable, its meaning is explicitly defined, it is linked to other external datasets, and it can in turn be linked to from external datasets"* [47].

The merger of the movement of Open Data with the concept of Linked Data gives raise to a powerful data organisation and knowledge distribution. The *Linked Open Data* (abbreviated LOD) as the combination of Open Data and Linked Data is a method of publishing machine-readable open data so that it can be interlinked among different datasets on the Web enabling data integration and semantic querying [47].

In the context of the Semantic Web, data should be available in Resource Description Framework (RDF triples) which also provides the possibility of querying the datasets using SPARQL. Data are also univocally identified by means of URIs and transferred through the HTTP protocol.

In 2010 Tim Berners-Lee proposed the *five-stars model* [40] which classifies Open Data into five different categories depending on the format on which data is distributed and is now widely accepted as framework evaluate quality of LOD projects. The five-stars classification schema is summarised in Table 1.1.

1.3 Ontologies

The word *Ontology* comes from the Greek *ontos* (being) and *logos* (study) and has its root in philosophy where it refers to the subject of being and existence as well as the basic categories [257]. In other words, the term Ontology is used to refer to *"the study of categories of things that exist or may exist in some domain"* [233].

Even though there is no universal definition for ontology, one of the most frequently cited in the Semantic Web literature is the one proposed by Gruber et al.: *"an ontology is a formal, explicit specification of a shared conceptualisation"* [117]. Here, *conceptualisation* stands for a simplified representation or an abstract model of the world within the domain considered; *shared* because it has to captures consensual knowledge (i.e., it is accepted by a group and not only by a single individual). Ontology is also an *explicit specification* which means that objects, concepts and relationships must be clearly defined; and *formal* indicates that the ontology should be machine understandable.

Ontology is also a well-known concept in artificial intelligence and in particular in the knowledge representation field. Knowledge engineers intend with ontology a means for representing knowledge in a way that machines can reason, that is, making inferences and valid deductions.

Uschold et al. highlight that an ontology, despite the several different formats it may assume, normally include a vocabulary of terms, specifying their meaning and indicating how they are interrelated [139]. More simply, an ontology is the representation of the knowledge according to a specific domain, where the concepts and their relationships are described by a vocabulary.

Within the context of the Semantic Web, ontologies categorise concepts into classes based on common attributes and characteristics reflecting the George Lakoff's "classical vision" of categorisation [56]. According to Lakoff's vision, a class is defined by a set of properties and the basic condition for an object to belong to a class is to possess all the properties associated with the class [151]. Properties may be defined as necessary and sufficient so that inference mechanisms will automatically identify membership.

Semantic Web ontologies enable machines to interpret and process information on the Web, providing a common model that can be understood both by humans and computer, to share, exchange, and reuse data based on their intended meanings.

The use of ontologies aims at achieving semantic interoperability by bridging and integrating multiple and heterogeneous digital content on a semantic level, which is exactly the core idea of the Semantic Web vision. Furthermore, not only the use of ontologies reduces the semantic ambiguities by offering a single interpretation resource, but also, information content is made available for machine consumption, whereas the majority of the content found on the Web today is primarily intended for human consumption only.

1.3.1 Ontologies Classification

In the literature, various different ontology classifications exist. As depicted in Figure 1.3, Guarino et al. propose a classification based on the degree of generalisation [120]:

- *Top Level Ontologies*: describe very generic and abstract concepts such as space, time, matter, object, event, action, etc. Ontologies of this kind are valid regardless of the specific problem or domain of interest.

- *Domain Ontologies*: describe a vocabulary related to a generic domain (e.g., medicine or a sport) by specialising the concepts provided by the top level ontology.

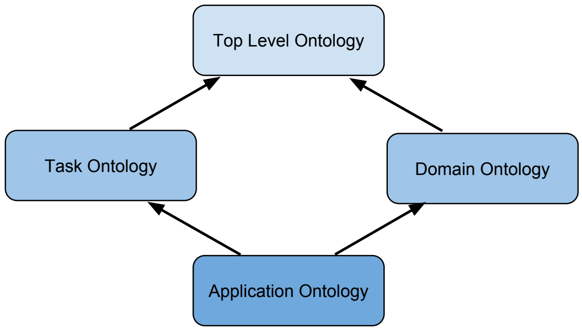

Figure 1.3: Guarino's ontology classification. Thick arrows represent specialisation relationships. (Image adapted from [120]).

- *Task Ontologies*: describe the vocabulary of terms needed to perform generic tasks or activities (e.g., diagnosis) by specialising the concepts provided by the top level ontology.

- *Application Ontologies*: describe the terms of concepts depending both on a particular domain and task. Ontologies of this kind are restricted only to a specific application.

McGuinness et al. propose a classification based on the internal structure of the ontologies; ontologies range from lightweight to heavyweight, depending on the complexity which characterises the elements they contain [169]. According to Corcho et al. a lightweight ontology is composed by concepts, properties, relationships and concepts taxonomies, while heavyweight ontologies are complex and include also axioms and constraints [73].

Gomez et al. suggest a classification which is partially orthogonal to the previous discussed above and it is based on the information represented by the ontology [113].

It is noteworthy to highlight that clear lines among these categories cannot be drawn, neither is there any formal specification to classify ontologies.

Strictly related to ontologies are the concepts of taxonomy and thesaurus. Even though taxonomies and thesauri are not specifically designed for the Web, in fact they don't appear on the Semantic Web stack, they, however belong to the Semantic Web picture.

Taxonomies

Daconta et al. define *taxonomy* as: *"the classification of information entities in the form of a hierarchy, according to the presumed relationships of the real-world entities that they represent"* [79]. A taxonomy provides a means to categorise, organise, label, and arrange information in hierarchical fashion using *father-son* relationships. A father-son relationship is a generalisation for the *is-a* and the *type-of* relationships, and is the only one kind of relationship which hold among concepts ruling out other relationships, such as *part-of*, cause-effect, association, and localisation. Furthermore, taxonomies do not permit defining attributes for terms.

Below an example of taxonomy; the classification of the human species in the Linnaean living being taxonomy[1]:

```
Kingdom: Animalia
        Filo: Cordata
            Subfilo: Verebrata
                Class: Mammalia
                    Subclass: Theria
                        Order: Primata
                            Suborder: Anthropoidea
                                Family: Hominidae
                                    Genera: Homo
                                        Species: Sapiens
```

Note that all the terms present are related by the generalisation relationship (e.g., *Mammalia* is a type of *Vertebrata*, which in turn is a type of *Chordata*, which in turn is a type of *Animalia*).

[1] https://en.wikipedia.org/wiki/Linnaean_taxonomy

Thesauri

According to the ANSI/NISO Monolingual Thesaurus Standard a *thesaurus* is defined as: *"a controlled vocabulary arranged in a known order and structured so that equivalence, homographic, hierarchical, and associative relationships among terms are displayed clearly and identified by standardised relationship indicators ..."*.

In other words, a thesaurus can be seen as a taxonomy together with a set of semantic relationships, such as equivalence, inverse, and association, that hold among the concepts.

A thesaurus can be used to guarantee that concepts are described consistently to enable users to refine searches and locate the information they need [56].

If relationships other than those thesauri support (i.e., equivalence, homographic, hierarchical, and associative relationships) are required, one must resort to more general ontologies.

A notable example of a thesaurus is WordNet[2]. WordNet is a thesaurus for the English language based on psycholinguistics principles and developed at the Princeton University by George Miller [176]. WordNet is an online lexical database designed for use under program control. English nouns, verbs, adjectives, and adverbs are organised into sets of synonyms, each representing a lexicalised concept. Semantic relations link the synonym sets [175].

1.3.2 Healthcare Ontologies

Due to the extreme complexity of medical terminology systems and medical information systems, ontologies play a central role for the representation, management, and sharing of knowledge and data.

Ontologies are preferred to conventional classifications due to the higher level of expressiveness that is possible to achieve in describing concepts

[2]WordNet is a registered trademark of Princeton University.

and their relationships. Furthermore, the domain knowledge in a machine processable format facilitates an efficient information retrieval.

In the past years, a plethora of healthcare domain ontologies have been created. Such representations are used to systemically denote, categorise, and relate healthcare data, allowing easier handling of the data in healthcare information systems [86].

Most of the existing healthcare ontologies are designed to describe a specific domain in biomedicine, such as the terms to describe anatomical parts and their relations, or terms used in clinical medicine, such as in EHR (Electronic Health Records) systems or rehabilitation domain [261].

Healthcare ontologies are widely recognised as a key factor technology to provide the semantics required for deriving proper treatment through integrating clinical guidelines [134].

The number of ontologies in the healthcare domain is constantly increasing; BioPortal provides access to a library of biomedical ontologies and terminologies developed in Web Ontology Language (OWL), Resource Description Framework Schema (RDFS), Open Biological and Biomedical Ontologies (OBO) format [256].

Below the main characteristics of SNOMED-CT and LOINC ontologies are briefly reviewed.

SNOMED-CT

The *Systematized Nomenclature of Medicine-Clinical Term*[3] (abbreviated SNOMED-CT) is considered as the main ontology for a standardised representation and automatic interpretation of clinical concepts, terms and relationships in the field of health care.

The ontology covers most of the areas that are used in medical practice, including clinical findings, symptoms, diagnoses, pharmaceuticals, body structures, medical devices, social contexts, and so on.

SNOMED-CT has hierarchy structure with a set of top level general concepts. All other concepts are subtypes of one these top concepts. Each

[3] `http://www.ihtsdo.org/snomed-ct`

concept is assigned a unique ConceptID and a Fully Specified Name (FSD).

SNOMED-CT provides a consistent way for indexing, storing, retrieving and aggregating clinical data that can enhance the interoperability between different health information systems.

LOINC

The *Logical Observation Identifiers Names and Codes*[4] (abbreviated LOINC) is a universal code system for laboratory test and other clinical observations. For each observation provides a code, a short name, a long formal name and synonyms.

The primary purpose of LOINC is to provide common codes and terminology which allow hospitals, pharmaceutical manufacturers, researchers, and public health departments to receive clinical observations from multiple sources, so that they can automatically file the data in the right slots of their medical records, research, and public health systems.

1.4 Reasoning

Reasoning is the process of extracting new knowledge (inferring facts that have not been explicitly stated) from an ontology and its instance base and is one of the most powerful features of Semantic Web technologies.

A *Semantic Reasoner* (also known as *reasoner engine* or simply *reasoner*) is a software system whose primary goal is to infer knowledge which is *implicitly* stated by reasoning upon the knowledge *explicitly* stated, according to the rules that have been defined.

The reasoners are also used to *validate* the ontology, that is, they check its consistency, satisfiability and classification of its concepts to make sure that the ontology does not contain any inconsistencies among its term definitions.

[4] `https://loinc.org`

According to Donini et al. the basic ontology reasoning procedures [91] can be listed as follows:

- *Consistency checking*: assures that the ontology does not contain contradictory facts (e.g., equality and inequality assertions).
- *Concept satisfiablility*: checks whether a class can have at least one individual or not. Having unsatisfiable classes usually means that the entire ontology is not consistent.
- *Concept subsumption* (*classification*): determines the subclass relationships between classes in an ontology in order to complete the class hierarchy.
- *Instance checking*: checks whether an individual is an instance of a class (i.e., it calculates the individual type).
- *Conjunctive Query Answering*: answers a (SPARQL) query with regard to an ontology.

As far as Description Logics (and Logics in general) are concerned, desirable properties of these reasoning techniques are:

- *Termination*: is related to guarantee that for a given input the algorithm can terminate.
- *Soundness*: ensures that every formula proved to be satisfiable, is indeed satisfiable.
- *Completeness*: concerns to the capability of deducing every possible fact that can be inferred from the available set of axioms.

A lot of research is currently being focused on investigating the compromise between the expressiveness of ontology definition languages and the computational complexity of the reasoning procedure, as well as the discovery of efficient reasoning algorithms applicable to practical situations [146].

Three classical open source reasoners available are: HermiT [223], Pellet [229] and FaCT++ [246].

1.4.1 Ontology Web Language

The Web Ontology Language (abbreviated OWL) [171] is an ontology language which extends RDFS to overcome its limitations. OWL is a W3C Recommendation and is the *de facto* standard for publishing and sharing ontologies in the Semantic Web.

RDFS is deliberately intended to be a simple language to define ontologies such as vocabularies and taxonomies but in many cases to address the demands of the Semantic Web more expressiveness is needed.

OWL as a markup language for specifications of ontologies has been used for applications in a large variety of fields such as medicine [111], biology [228], agriculture [231] and defence [149].

OWL mainly derives from DAML+OIL Web Ontology Language [170] [131] which in turn is a combination of DAML [127] and OIL [98].

Like RDF Schema, OWL can be serialised using RDF syntax and adopts the *open world assumption* (which means that missing information is treated as unknown) and the *not unique name assumption* (different identifiers may refer to same entities in the real world).

OWL introduces many new language primitives which extend RDF and RDFS. OWL allows to define classes as a combination of other classes using set operators like union, intersection and complement. In OWL is possible to state that two classes are disjoint or are the same (despite being identified with different URIs). It is also possible to use restrictions on properties such as *cardinality* or specify that a certain property is *transitive* or *unique*.

The Web Ontology Language provides richer schema for expressing meaning and semantics but the more expressive is a language, the more is difficult to reason with the language. Although complex language constructs allows to represent more knowledge, computation becomes inefficient and eventually undecidable.

When it comes to choosing and ontology language for the Semantic Web there is always a trade-off between expressibility and efficient reasoning, depending on the kind of application to be designed.

OWL consists of a family of three languages with different degrees of expressivity and computational properties: OWL Full, OWL DL and OWL Lite.

- *OWL Full* is the most expressive language it places no restrictions on how the language constructors can be used. This flexibility comes at the expense of decidability, in fact reasoning task such as consistency checking, satisfiability checking, subsumption checking, instance checking and conjunctive query answering. All of these typical reasoning tasks over an OWL Full ontology are undecidable.

- *OWL DL* and OWL Lite are two restricted forms of the OWL language, restrictions make them "decidable". Both OWL DL and OWL Lite are based on *Description Logic* (abbreviated DL) [27] which guarantees (all conclusions are guaranteed to be computable) and decidability (computation will be finished in finite time).

  OWL DL is the more expressive after OWL Full and is also the most important among the three variants of the OWL family. OWL DL is equivalent to a well-defined DL and contains all of the OWL language primitives but allows restricted use of them. A full list of restrictions put in OWL DL can be found in [171].

  OWL DL is decidable for consistency, satisfiability and instance checking tasks. However the complexity of these reasoning task are NExpTime-complete which means that for certain valid inputs the reasoning task may not be completed in "acceptable time".

- *OWL Lite* is a subset of OWL DL and it is most restricted variant of OWL. The rationale behind OWL Lite is to trade expressivity for efficiency of reasoning: *"reasoners for OWL Lite will have desirable computational properties"* [255]. The complexity of OWL Lite is

ExpTime-complete for consistency, satisfiability and instance checking tasks. As opposed to OWL DL, conjunctive query answering is decidable, however OWL Lite 2ExpTime-complete with respect to query complexity [77] which means that for certain valid inputs, despite the certainty of decidability, reasoning is intractable.

OWL 2

OWL 2 [129], addresses in part the issues which afflict the previous version of the language and introduces new language primitives and semantics for OWL 2 Full and OWL 2 DL. A comprehensive report of the rationale and new features introduced bye OWL 2 can be found in [110], here is given a brief overview.

While OWL 1 defines only two main dialects OWL Full and OWL DL one syntactic subset (OWL Lite), OWL 2 provides in addition three new *profiles*: OWL 2 EL, OWL 2 QL, and OWL 2 RL. These profiles are syntactic subsets of OWL 2 DL and are intended to target different application scenarios by trading the expressivity to achieve an efficient reasoning.

- *OWL 2 EL* is based on the Direct Semantics [180] and it is primarily designed for dealing with a large number of class axioms and classification tasks (such as subsumption and instance checking).

  OWL 2 EL was conceived to address the complexity of numerous existing large-scale ontologies in the healthcare and life sciences domain such as SNOMED-CT (an ontology of clinical terms with over 500000 classes). [230] or Gene Ontology (a biological ontology that describes genes and gene properties with more than 25000 classes)[70].

  Reasoning for OWL 2 EL is PTime-complete (polynomial complexity) except for query-answering [180].

- *OWL 2 QL* is also based on the Direct Semantics and provides more expressive features such as the property inclusion axioms and functional and inverse-functional object properties.

The QL profile of OWL2 was developed to efficiently handle query answering in ontologies which contain a large number of individual assertions and relatively uncomplicated class definitions. OWL 2 QL also adopts technologies from relational database management.

Reasoning is NLogSpace-complete with the exception of query answering which is NP-complete [180].

- *OWL 2 RL* is based on Description Logic Programs (DLP) as proposed by Grosof et al. [114] and pD* proposed by ter Horst et al. [242].

  OWL 2 RL enables interaction between description logics and rules, in fact it was primarily designed to deal with ontologies that describe rules within. OWL 2 RL is basically a rule language and rules can efficiently be run in parallel, allowing for scalable reasoning implementations.

  Reasoning in OWL 2 RL is PTime-complete except for query answering which is NP-complete [180].

1.4.2 Semantic Web Rule Language

The Semantic Web Rule Language (SWRL) is a rule language, fully compliant with OWL semantics, that is meant to combine OWL knowledge bases with Horn-like rules in order to extend reasoning capabilities [181, 130].

SWRL rules are composed of an implication between an antecedent (also referred to as the rule body) and consequent (or head). When the conditions specified in the antecedent are true also the conditions specified in the consequent must hold. Both the body and the head are positive a conjunction of atoms.

Atoms consist of a predicate followed by a number of terms or arguments of the expression in the form:

```
p(arg1, arg2, ..., argn)
```

Five types of predicate can be used in rules: OWL classes, OWL properties, data types, data ranges, and built-ins. Arguments can be either OWL individuals or data values or variables which are treated as universally quantified.

A rule asserting that the uncle of an individual is the individual's father's brother can be written as:

```
Person(?x)
^ hasParent(?x,?y)
^ hasBrother(?y,?z)
-> hasUncle(?x,?z)
```

The execution of this rule has the effect of setting the property hasUncle to the individual ?z in the individual ?x. That is, the individual that satisfies the antecedent of the rule.

The highest expressivity level provided by SWRL can be reached through the use of built-ins [10]. Specifically, a built-in is a predicate that evaluates to true if the arguments satisfy certain conditions. For example, the built-in `equal` accepts two arguments and return true if the arguments are the same. Built-ins can perform a wide range of task from mathematical operations to string manipulations. Moreover, users can implement their own customised built-ins.

For example, built-ins can be used to determine whether a string representing a telephone number starts with the international access code "+" as in the following rule:

```
Person(?p)
^ hasNumber(?p, ?number)
^ swrlb:startsWith(?number, "+")
-> hasInternationalNumber(?p, true)
```

SWRL, due to the full OWL semantics compliance, adopts the open world assumption and does not allow to assume that two individuals are distinct on the sole base of their names (unique name assumption). Moreover, SWRL supports monotonic inference only, that is rules cannot retract or modify existing information in the knowledge base.

The increased extent of expressivity introduced by SWRL can potentially lead to undecidabiliy. DL-Safe SWRL rules are a restricted subset of SWRL rules that ensure the desirable property of decidability which is obtained by restricting rules to operate only on known individuals in the knowledge base [181]. However, even though all the deductions obtained through DL-Safe rules are formally sound, they may be incomplete.

1.5 Semantic Data Annotation

To achieve the Semantic Web goal of making machines able to interpret, combine and use information on the Web, data need to be semantically annotated.

According to Amardeilh, *Semantic Annotation* is defined as: *"a formal representation of content, expressed using concepts, relations and instances as described in an ontology, and connected to the original resource"* [23].

Within the Semantic Web context, ontologies play a central role in annotation tasks since they explicitly define concepts and relations among them of a particular domain, in a structured and formal way.

Annotation is essentially the process of adding *metadata* to data. Metadata are *"data about data"* [125] and are normally structured according to an ontology, which means that their values refer to the instances and concepts defined in the ontology.

Consequently, semantic annotation turns human understandable content into a machine understandable form by enriching data with metadata to ensure machine readability.

It is noteworthy to underline that metadata alone without being associated to an object are meaningless.

Semantic annotation can virtually be applied to any kind of resource such as textual resources, web pages, images, multimedia contents, fields in databases and numerical data [144].

Annotations can be *embedded* or *detached*. Embedded annotations are directly added within the resource's content. Instead, detached annotations are stored outside the resource's content.

Finally, it is important that the process of semantic annotation adheres to a common standard to guarantee interoperability between different systems. The Resource Description Framework (RDF), the cornerstone of the Semantic Web, provides a standardised means for adding metadata annotations to resources.

According to Lefrançois et al. *"RDF data model may still be used as a lingua franca to reach semantic interoperability and integration and querying of data having heterogeneous formats"* [156]. Therefore, generating RDF triples (*triplify*) from sources having various formats is a key step for every Semantic Web system.

1.5.1 Sources Heterogeneity

Sources Heterogeneity refers to when within a single domain, heterogeneous formats express homogeneous content. That is the same concepts are represented using different types and stored using a multitude of data models and formats.

A brief survey of different solutions that have been proposed for generating RDF models from data in heterogeneous formats and serialisations can be found in [87] and [88].

Dimou et al. identified some limitations of the existing mapping methods (data-to-RDF) which prevent achieving well integrated datasets: *mapping of data on a per-source basis*, *mapping data on a per-format basis* and *mapping definitions' reusability* [88].

In particular, mappings tools based on a *per-format* approach only support a specific source format (e.g, XML) which leads to a proliferation of tool to install, learn, use and maintain for each case separately or even to implement *case-specific* solutions.

Furthermore, often the mapping rules are not interoperable because they are tightly coupled to the implementation. In this case, it is not possible to reuse the mapping rules to map data that describe the same model, for different data serialisations.

Dimou et al. also proposed the requirements for generic mapping systems to address the aforementioned issues and achieve a better integration which are as follows: *uniform and interoperable mapping definitions*, *robust cross-references and interlinking* and *scalable mapping languages* [88].

In particular, the uniform and interoperable mapping definitions factor, requires the mapping definitions to be independent from the references to the input data. The same mapping definitions (i.e., mappings that capture the same concepts) should be available to be reused across different sources only by changing the reference to the specific values in the input source.

IoT Data Formats

In a context such as the Internet of Things, due to the large diversity of devices, data sources come in very large volumes and can be very heterogeneous in terms of serialisations and data formats. For instance, IoT data generated by fitness tracking devices can be normally retrieved in tabular-structured format such as CSV or hierarchical-structured format such as XML or JSON.

This section briefly describes two of the current most common data formats in the IoT fitness domain: XML and JSON.

The *eXtensible Markup Language* (abbreviated XML) is a W3C Recommendation and a markup language for encoding data in semi-structured format [55].

XML is a metalanguage and it does not define a predefined set of tags, rather it can be used to create markup languages for specific specialised domains and purposes by specifying tags and the relationships among them.

An XML document is represented as an ordered labelled tree according to the DOM standard [258] where each node in the tree corresponds to an element and may have a value, attributes, and namespaces associated. Leaf nodes normally contain textual data values. An XML document may also carry additional element such as comments, document level information (e.g., DTD - the document type declarations), processing instructions, entities and notations.

Several XML query and processing languages are proposed and recommended by W3C such as: *XPATH* [38], *XSLT* [68] and *XQUERY* [72].

XPATH which stands for *XML Path Language* is an expression language used for navigating and selecting specific nodes within an XML document. XPATH cannot create new nodes or modifying the existing document.

Below is shown an example of a typical XML document:

```
<?xml version="1.0" encoding="UTF-8"?>
<contacts>
   <contact>
      <name>Phil Clarkson</name>
      <phone>123-456-7890</phone>
      <mobile_phone>222-654-5432</mobile_phone>
      <company>Planetgreen</company>
   </contact>
   <contact>
      <name>Adrian Vance</name>
      <phone>765-178-8236</phone>
      <company>Biolam</company>
   </contact>
   <contact>
       ...
    </contact>
       ...
</contacts>
```

An example of XPATH expression to retrieve the phone numbers of all the contacts stored could be as follows:

```
/contacts/contact/phone
```

The output returned by the XPATH expression above:

```
<phone>123-456-7890</phone>
<phone>765-178-8236</phone>
```

Note that even if the first contact has two phone numbers associated, a fixed phone number and a mobile phone number, the latter is not retrieved due to the purely syntactic approach of querying the XML tree.

The *JavaScript Object Notation* (abbreviated JSON) is a lightweight, text-based, data interchange format [76]; is much simpler than XML and has a human-readable syntax and self-describing.

JSON was initially intended to be used in the JavaScript scripting language but then it did evolve into a language-independent data representation and it is supported by a wide range of programming languages.

JSON is essentially based on two data structures: *objects* and *arrays*. Objects are an unordered collection of name-value pairs, while arrays are an ordered list of values. JSON supports four data type which are as follows: strings, numbers, boolean expressions and null values. These features allow JSON to describe any kind of resource.

Compared to XML, JSON has higher parsing efficiency, a lighter syntax (XML is extremely verbose) and it is easier to read by humans.

Similarly to XPATH which is used to extract data from and XML document, *JSONPATH* is a declarative query language for selecting and extracting values from a JSON document [109].

The example below show the same document proposed in Section **??** serialised in JSON format:

```
{
   "contacts":{
      "contact":[{
         "name":"Phil Clarkson",
         "phone":"123-456-7890",
         "mobile_phone":"222-654-5432",
         "company":"Planetgreen"
        },
```

```
        {
         "name":"Adrian Vance",
         "phone":"765-178-8236",
         "company":"Biolam"
        },
        {
           ...
        }]
    }
}
```

1.5.2 RDF Mapping Language

The *RDF Mapping Language* (abbreviated RML) is a generic mapping language which allows to map heterogeneous data sources into RDF representation [88].

From a language point of view, RML extends R2RML (RDB to RDF Mapping Language) which is a W3C recommendation for expressing customised mappings from relational databases to RDF, according to a structure and vocabulary defined by the mapping user [82].

RML, like R2RML, is a triple-oriented mapping language and can be expressed as RDF graphs and written down in Turtle syntax. However, while R2RML is specifically designed to address relational databases, RML extends this scope to a broader set of different input sources data structures and serialisations (such as CSV, XML, JSON, etc).

The main limitation of R2RML is indeed that R2RML can deal only with relational databases input.

RML, while maintaining backward compatibility with R2RML, provides a generic way for defining mappings over a wide set of heterogeneous sources adding case-specific extensions.

Given that RML, unlike R2RML, deals with different data serialisations, specific query languages are needed to refer to the content of a specific resource (e.g., XPATH for XML files or JSONPATH for JSON files).

Sources of the same domain which adapt to different structures may represent the same information and RML mapping definitions can also be re-used across them with minimal modifications and combined in a uniform way to incrementally form a well-integrated resulting dataset.

Below the main structure of RML mapping graph is shortly described.

An RML mapping consists of one or more *Triple Maps*. A Triple Map is composed of three parts: (1) the *Logical Source*, (2) the *Object Map* and (3) zero or more *Predicate-Object Maps*.

The Logical Source extends the concept of a R2RML's *Logical Table* and it is used to determined the input source data to be mapped.

Reference Formulations (`rml:referenceFormulation`) are the means by which it is specified which standard or query language is used to refer to the data. The predefined Reference Formulations of the current RML version (at the time of writing) are: `ql:CSV`, `ql:XPath`, `ql:JSONPath` and `ql:CSS3`.

Unlike R2RML in which *per-row* iterations occur through the table data, the iteration pattern in RML has to be specified according to the data source format. The *Iterator* `rml:iterator` allows to define the iteration pattern over the input source and specify the extract of the data to be mapped during each iteration.

Similarly to the R2RML's property `rr:column` which defines a *column-valued term map* to determine a column's name, in RML is introduced the `rml:reference` property to reference to the single parts of the data input.

Both the iterator's value and the reference's value have to be expressed in a valid expression according to the Reference Formulation defined in the Logical Source.

The *Subject Map* (`rr:SubjectMap`) defines the criterion by which unique identifiers (URIs) are generated for the resources to be mapped. The same URIs are also used as the subject for each RDF triple produced from the Triple Map.

The *Predicate-Object Map* consists of a *Predicate Maps* and an *Object Maps*, which respectively generate the predicates and the objects for the subject generated by the Subject Map.

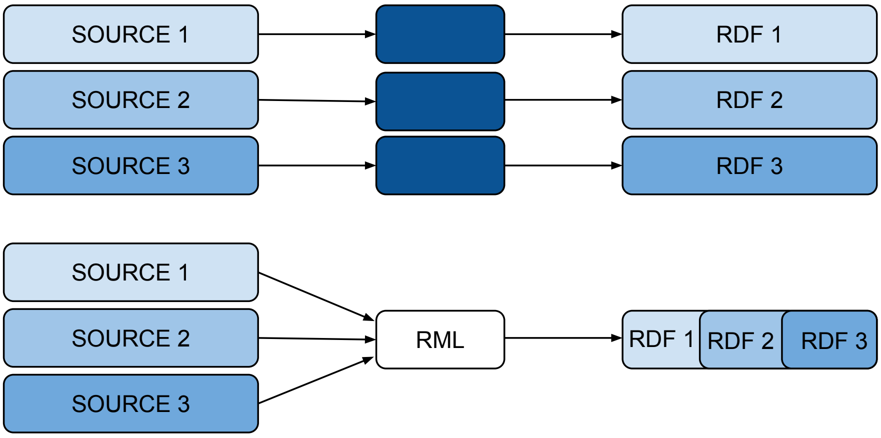

Figure 1.4: Mapping without and with RML. (Image adapted from [88]).

RML allows also cross-references through *Referencing Object Maps* which acts like a join operation between different mappings. A Referencing Object Map links together the values produced by a subject map (the parent map) to the objects of triples produced by another map (the child map). The join conditions are specified by the properties `rr:parent` and `rr:child`.

RML Mapping Process

This section shortly reviews the RML mapping process and proposes a simple example of an XML-to-RDF mapping. Executing an RML mapping requires an input source and a mapping specification that describes the `TipleMaps` and points to the input source.

According to the mapping specification document, the RML processor applies the mapping rules specified in the `TipleMaps` (the `Subject Map` and the `Predicate Object Maps`) to the input data. For each point of reference to the data within the input source, values are extracted by evaluating the corresponding *target expressions* and the triples are generated.

The resulting RDF graph can be stored in a user-defined format.

Below the RML mapping definition document serialised using the Turtle syntax:

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

<#ContactsMap>
rml:logicalSource [
    rml:source "contacts.xml";
        rml:referenceFormulation ql:XPath;
        rml:iterator "/contacts/contact";
];
rr:subjectMap [
        rr:termType rr:BlankNode;
        rr:class foaf:Person;
];
rr:predicateObjectMap [
        rr:predicate foaf:name;
        rr:objectMap [ rml:reference "name" ];
];
rr:predicateObjectMap [
        rr:predicate foaf:phone;
        rr:objectMap [ rml:reference "phone" ];
] .
```

The corresponding input source `contacts.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<contacts>
   <contact>
      <name>Phil Clarkson</name>
      <phone>123-456-7890</phone>
```

```
    <mobile_phone>222-654-5432</mobile_phone>
    <company>Planetgreen</company>
  </contact>
  <contact>
    <name>Adrian Vance</name>
```

```
    <phone>765-178-8236</phone>
    <company>Biolam</company>
  </contact>
</contacts>
```

The RDF output graph produced:

```
_:4evc1bCWsX a foaf:Person ;
  foaf:name "Phil Clarkson" ;
  foaf:phone "123-456-7890" .

_:8yNzIbnRMw a foaf:Person ;
  foaf:name "Adrian Vance" ;
  foaf:phone "765-178-8236" .
```

In this example, data stored in the `contacts.xml` file have been semantically annotated according to the *FOAF ontology definitions* [58] and serialised in Turtle syntax.

Note that values extracted from input sources may not always be in the correct form to be directly inserted in RDF triples. RML does not provide any means for *data cleansing* and according to Dimou et al., data cleansing if necessary should be performed in advance [88]. Heyvaert et al. propose a case in which they address the problem by extending the RML vocabulary (with the terms: `rml:process`, `rml:replace` and `rml:split`) to further process the values extracted using the *regular expressions* [128].

Chapter 2

Internet of Things

This chapter introduces the Internet of Things technologies, in particular the role of the Internet of Things in the healthcare and smart home domain. Critical aspects of IoTs such as interoperability issues, from a data-centric perspective, are taken into a detailed consideration. Secondly it offers an overview of the most common IoT fitness and domotic devices available on the market.

2.1 IoT

The term Internet of Things (IoT), sometimes also referred to as Internet of Objects or Smart Objects, denotes any combination of hardware and software that produces data through connecting multiple devices and sensors to the internet. The term Internet of Things was first introduced by Kevin Ashton at the Auto-Id centre of the Massachusetts Institute of Technology (MIT) back in 1999.

Anything can be an IoT device if it can transmit and receive data over the Cloud or, in other words, any system that can connect objects or things to Internet, hence connecting the physical world to the virtual world. Internet as a medium to communicate and exchange information is a living entity, constantly changing and evolving, and now is shifting from only connecting

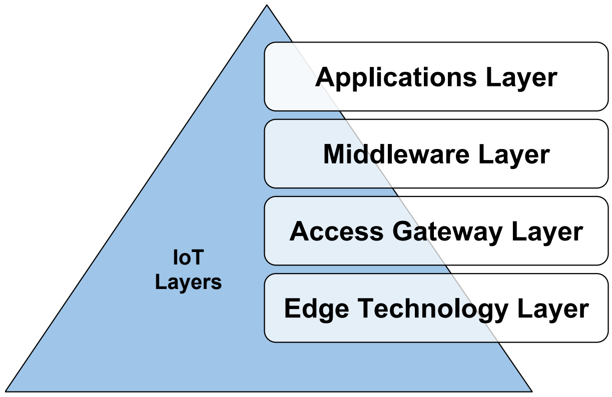

Figure 2.1: Layered Architecture of an IoT System. (Image adapted from [221]).

people and computers towards connecting things and objects.

This vision where objects become a part of the Internet is also possible due to an unceasingly evolving technology: Internet broadband connectivity is becoming cheaper and ubiquitous, devices are becoming smaller and more energy efficient and fitted with a large variety of on-board sensors. IoT is nowadays recognised as one of the technologies that will radically and permanently transform our life, business, and the global economy in the near future [164].

IoT paradigm can be applied to a long list of different domains ranging from transportation, supply chain, environmental monitoring, inventory and production management, smart cities, smart homes, building automation, data collection, social networks to medical care, healthcare. These latter ones in particular represent one of the most attractive application areas for IoT [191]. More and more of new applications and businesses for IoT are created continuously.

As shown in Figure 2.1, Santucci et al. describe the architecture of a typical IoT device as a four layered architecture: the edge technology layer, the access gateway layer, the middleware layer and the application layer

[221]. The two lowest layer are responsible for data collecting and network connectivity while the two highest layers are responsible for data utilisation in applications. The functions of every single layer (from the lowest to the highest) are as follows:

1. *Edge Technology Layer*: this layer is also known as perception layer is the hardware layer which includes components for network connectivity, data storage and data collection through sensors such as GPS, cameras, pressure sensors, temperature sensors etc. The Edge Technology Layer also provides information processing (via embedded edge processor), control and actuation.

2. *Access Gateway Layer*: this layers is also known as network layer or transport layer and is responsible for data transmission and routing. It receives information from the edge layer using communication technologies such as Wi-Fi, Li-Fi, Ethernet, GSM, WSN, ZigBee Bluetooth and WiMax [221, 26] and sends them do the middleware.

3. *Middleware Layer*: this layer provides an abstraction to applications from things. It also provides services such as data filtering, data aggregation, semantic analysis and access control.

4. *Application Layer*: which is also the top layer of the stack consists of two sub-layers: the data management sub-layer and the application service sub-layer. The data manager sub-layer provides directory service, quality of service (QoS), cloud computing technologies, data processing, machine-to-machine (M2M) services etc. The application service sub-layer on the other hand is responsible for interfacing the system to end users and enterprise applications running on top of the IoT applications layer.

Information processing is handled in application layer. The information processing technologies for IoT applications include also Cloud Computing and Fog Computing. For instance, in the case of healthcare applications that depend on utilisation of inputs from the physical world (e.g., vital signs of a patient via sensors), a huge amount of data is constantly collected. The data can be sent to a cloud integrated with the IoT system for a safe, convenient and efficient storage, processing and management [92]. The cloud-based approach enhances healthcare solutions by improving accessibility and quality of healthcare, and reducing costs [107].

Similarly, fog computing extends cloud computing. It is a distributed computing infrastructure that provides the same application services to end-users as cloud computing such as data processing, storage, and execution of applications. However, the application services are handled at the network edge in a smart device instead of a remote datacenter in the Cloud. The goal of fog computing is to improve the efficiency and reduce the amount of transported data to the Cloud [51].

2.2 Critical Issues in IoT

Despite the growing number of IoT devices and applications, IoT technology is still in its infant stage and has big room for research in variety of issues such as standards, scalability, heterogeneity of different devices, common service description language, safety and integration with existing IT systems just to cite a few.

Interoperability as the ability to interconnect and communicate different vendors' systems along with data integration is one vital issue still unsettled.

Barnaghi et al. highlight four interoperability issues in IoT [30]:

1. *Technical interoperability* involves the heterogeneity of hardware and software components and the related communication protocols.

2. *Syntactical interoperability* involves data formats and data representation. Syntactical interoperability is crucial to interpret IoT data and build smart systems. They underline the need to agree on common vocabularies to describe data.

3. *Semantic interoperability* involves the interpretation of meaning of data exchanged.

4. *Organisational interoperability* involves the heterogeneity of the different infrastructures. Organisational interoperability depends on successful technical, syntactical and semantic interoperability.

Nowadays the majority of IoT applications tend to be self-contained thereby forming application silos [250]. Chen et al. state : *"cannot correlate and integrate the data from different silos and getting the data from heterogeneous sources"* [66]. The authors highlight the needs for IoT data processing and explain the issue related to domain specific-applications: applications cannot combine the data from different silos.

Sensor data are useless if are not analysed and understood correctly. Interpreting raw IoT data, extracted from devices, in a meaningful way is still an open issue and a challenge [104].

Interoperability can be solved if communicating smart systems are semantically interoperable [105]. Semantics gives a structure to data and captures the meaning. This challenge is particular relevant in the health care and fitness domain where a multitude of diverse vendor devices collect the same type of data but store and exchange them in many different ways, so there will be semantic and syntactic conflicts. Semantic Web technologies are promising tools for this purpose to share data and exchange their services efficiently [138]. Semantic Web technologies are also the approach that has been adopted in this study.

2.3 Web of Things

The main goal of IoTs is to connect physical devices to the Internet. The concept of Web of Things (WoT) [260] concerns the connection of the sensors specifically to the web, getting the data and exchanging the data ,that has been produced by devices, on the web.

Existing web technologies can be adapted and reused to build new smart applications and services exploiting data generated by the IoT devices by integrating Smart Things to the Web. Web services have been proven to be crucial in creating interoperable applications on the Internet, IoT devices can be abstracted as web services and seamlessly integrated into the existing web. The WoT vision depicts a view where a collection of web services can be discovered, composed and executed.

There are two possible methods to integrate things into the Web: direct integration and indirect integration [121]. In many cases IoT systems use both methods has a hybrid way.

Direct integration means integrating things into the Web using embedded web servers. IoTs running an embedded web server can directly communicate with the users from any terminal with a standard web browser. Other devices can also inter-operate with them through standard web operations specified by web standards (e.g., GET and POST). Web severs can be built in a size of only a few KBs [93] [17] so that they can be easily embedded into many devices directly despite the limiting memory and computational capabilities of the IoT devices. Indirect integration on the other hand is needed when a device is not powerful enough to be embedded with a web server. Sometimes there is also no need to directly integrate all the smart things to the Web in the consideration of cost, energy and security[121]. For indirect integration, an intermediate proxy (also called smart gateway) placed between things and the Web is used. The proxy communicates directly with the smart things, this implies that it understands the proprietary protocols of the devices, and exposes outward to the Web the proprietary protocols and the native APIs of the smart things abstracting them. In this way IoT can still be accessible using web standards.

2.3.1 Semantic Web of Things

Semantic Web of Things (SWoT) is a research field which aims to combine Semantic Web technologies to Internet of Things providing interoperability among ontologies and data [138, 194].

Existing WoT systems deal with heterogeneous protocols and easily share sensor data on the Web. However, they do not use Semantic Web technologies. SWoT differs from WoT by adding semantics in order to ensure a common understanding. Semantic Web of Things can be seen as an

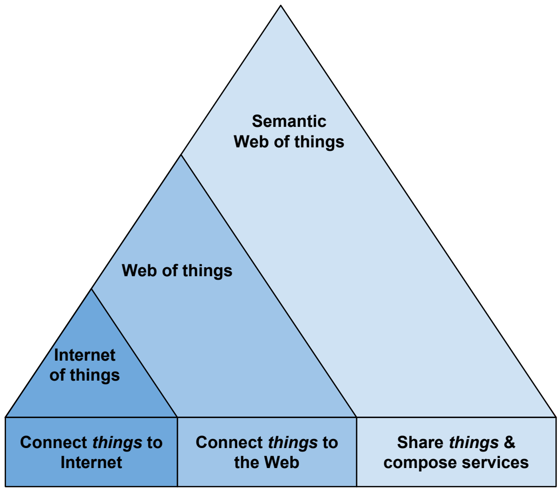

Figure 2.2: From IoT to SWoT. (Image adapted from [138]).

evolution of Web of Things through integration of IoT with web technologies to access the devices and the produced data via Web.

Barnaghi et al. show that semantic is needed to: (1) provide unambiguous IoT data descriptions to be interpreted by machines, (2) combine data from different physical systems and devices, (3) semantic reasoning, and (4) sensor discovery [31].

SWoT promises a seamless extension to the IoT allowing integration of both the physical and digital worlds and are focused on providing wide scale interoperability that allows the sharing and reuse of data [138]. The SWoT vision enables also knowledge-based systems to achieve high degrees of autonomic capability for information storage, management and discovery leveraging on ontologies and standardised semantic web languages such as RDF, RDFS and OWL.

2.4 IoT and Healthcare

The healthcare industry has seen a radical change in the era of the Internet with relatively cost efficient and smart solutions such as IoT technologies.

According to Atzori et. al the healthcare domain has a huge potential for IoT successful applications and smart systems [26].

Koop et al. envision a new delivery model of healthcare, enabled by the IoT technology, that will transform the present hospital-centric, through hospital-home-balanced by 2020, to the final home-centric by 2030 [147]. Healthcare providers around the world are transforming themselves into more efficient, coordinated and user-centred systems and technology plays a central role in achieving efficiency and enhancing distributed healthcare smart systems that fulfil diverse and constantly increasing demands.

Significant segments of the IoT healthcare market are: independent living services, consumer medical devices, telemedicine, wearable technologies, fitness monitoring devices, health gaming, personal emergency response systems and wearable technologies [99].

Typical IoT solutions for healthcare can be categorised as followings:

- *Tracking and monitoring*: thanks to the ubiquitous identification, sensing, and communication technologies patients and people can be tracked and monitored by wearable devices on a 24/7/365 basis [21]. Wearable fitness tracking devices and life logging devices belong to this category.

- *Remote service*: healthcare and home assistance, emergency detection and first aid, dietary and medication management, telemedicine and remote diagnosis can be delivered remotely through the Internet by connected devices [196][145][158]. Remote monitoring of patients allows more self-management of chronic conditions, and significant services improvements and cost reductions.

- *Information management*: enabled by the global connectivity of the IoT, all the healthcare information (logistics, diagnosis, therapy, recovery, medication, management, finance, and even daily activity) can be collected, managed, analysed and utilised throughout the entire value chain [90].

- *Cross-organisation integration*: the hospital information systems (abbreviated HIS) are extended to patient's home, and can be integrated into larger scale healthcare. IoT technologies facilitate the flow of patient data throughout an expanding community of care (medical centres, hospitals, nurses, physicians and associated systems) while also securing the information and protecting patients privacy [99].

The healthcare sector is just one of the markets that IoT will transform in the coming years. IoT will radical modify our medical system by bringing technology directly into the home, changing the way healthcare is delivered to patients and consumers. Moreover, with billions of heterogeneous sensors accumulating a robust network of data collection and data sharing coupled with ubiquitous identification systems, experts can conduct real-time data mining and interpretation, which leads to a continuous quality improvement to the sector.

IoT technology already offers a multitude of networked devices, cloud based applications and services for healthcare. Disparate types of healthcare data and information like logistics, diagnosis, recovery, therapy, meditation, management, finance and even daily activities (e.g., through wearable devices) can be collected from the IoT systems [90][78]. In short, more connections mean more accessible data and better healthcare for patients.

An example of the vision described above is the Electronic Health Record (EHR) which is already adopted in various countries of the world [236]. Electronic Health Record is the collection and digitally storing of health information about individual's lifetime with the purpose of supporting continuity of care, education and research [132].

Healthcare IT companies are also developing Personalised Health Records (PHR) where users can collect and update their facts [25], this process can be sustained and automated, by mobile based systems and IoT. IoTs automatically update physical activity, vital symptoms and similar information.

2.4.1 IoT Fitness Devices

In recent years, the consumer market of IoT healthcare devices has seen a proliferation of wearable fitness trackers such as Fitbit[1] and Jawbone UP[2] or smartwatches like Apple Watch[3]. These devices, along with other functions, provide a lot of health tracking features. With the rising of wearable devices people are becoming more and more interested about their health and IoT health trackers devices are becoming part of normal daily life. According to a survey conducted by the Intercontinental Marketing Services Institute for Healthcare, the sales of wearable technology will grow to almost US $30 billion by the next year[4].

IoT fitness devices can record the exercise amount, consumed energy, food intake and sleeping status of users per day. They can also measure various physical indexes such as heartbeat and respiration rate, monitor their data including speed and running distance. Sometimes they also provide support for improving exercise and goal achievements such as weight loss or distance travelled.

Wearable devices are a great tool for collecting biometric data in real time for an extended period of time to manage and prevent some chronic disease [173]. Fitness trackers are almost wearable devices such as smart wristbands, heart rate strip and smart wristwatches. Some of the same functionality and sensors are also present in modern smartphones [237].

IoT fitness devices can realise exercise step counting, exercise tracking, heart rate counting, sleeping tracking as well as real-time exercise and sleeping monitoring, diet tracking, smart alarm clock, customised alarm, emotional tracking, distance course, step collection, calorie burning measurement, sleeping quality, motion reminder, smart no-sound alarm clock, distance counting and measuring calorie consumption [159, 238].

Along with wearable devices, mobile phone health apps are changing the healthcare by empowering users and educating them to take control

[1] https://www.fitbit.com
[2] https://jawbone.com/up/trackers
[3] https://www.apple.com/watch/
[4] https://www.webcitation.org/6cxkgjwZu

and track of their health and their fitness gains.

Wearable Devices

Fitness trackers have become increasingly popular during recent years. Wearable devices for fitness tracking and health monitoring consist mostly of smartwatches and wristbands. However, there are also fitness trackers that can be worn on the shoes, on the waist or on the upper arms.

Wristbands

A typical wrist-worn device collects and sends data such as the wearer's step count or wearer's heart rate through a gateway (e.g., a connected device like a smartphone) to the company's server.

Research has shown that data collected by these devices, despite being noisy and sometimes inaccurate, can even be used to answer intimate questions, such as whether two persons are working together (by tracking the similarity of steps per minutes between users) [247] or if the wearer has recently quit smoking [142].

All fitness trackers available in the market are equipped with an accelerometer sensor and achieve a common core functionality which is the step counting. The accelerometer sensor alone is used to infer a lot of user activities during the day such as number of steps taken, calories burned, distance travelled, as well as time slept during the night.

Wristband based accelerometers also known as *actigraphs*, such as the basic models of Fitbit or Jawbone UP, are one of the most commercially successful types of wearable sensors. Their success id due to the fact that they are cheap and can detect a wide range of daily activities (e.g., sleep, household chores, and various forms of exercises) [69][204].

Multi-sensors wristbands, as more sophisticated models of wristband, in addition to the accelerometer are also equipped with localisation services such as GPS and sensors for measuring heart rate, body temperature and blood oxygen levels (e.g., through an infrared sensor).

Data collected by wristband devices can be transmitted wirelessly for real-time feedback or uploaded to the cloud even though some basic models sync only when physically connected to a computer via cable. User interfaces of wristband devices are very limited, normally these devices are provided with only a single-button and sometimes a tiny display to show some basic information.

Smartwatches

A smartwatch is a wrist-worn, besides being also a timekeeping device, *"general-purpose, networked computer with an array of sensors"* [205].

Smartwatches allow more computational capabilities (actually the major part of smartwatches in the market are wearable computer) than the traditional fitness bands and host a lot of more accurate bio-sensors.

Modern smartwatches are fitted with sensors like: tri-axial accelerometers, gyroscopes, microphones, ambient light sensors, optical sensors, wireless signal strength and GPS systems.

Smartwatches' fixed mount location on the body and continual connection to the skin makes them capable of recognising wearer's physical activities with a high degree of precision. The device collocation also permits easy recording of heart rate, heart rate variability, temperature, blood oxygen, and galvanic skin response (GSR).

Reeder et al. state that smartwatches have the potential to transform the healthcare by constantly monitoring the users daily. In particular they highlight the following points of strength of smartwatches: (1) are familiar to most people, (2) are increasingly available as a consumer device, (3) enable near-real time continuous monitoring of physical activity and physiological measures, (4) support messaging and reminders, (5) enable communication between patients, family members, and healthcare providers, and (6) allow for *in situ* mini-surveys and behaviour verification based on sensors measures [209].

Some limitations of smartwatches are the physical characteristics of the wearable devices, such as the small screen size which impaired the usability

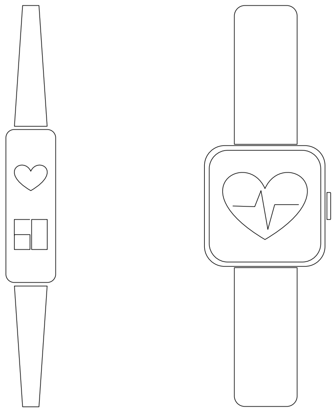

Figure 2.3: A Wristband and a Smartwatch.

of the device [29] and the energy consumption [159] which affects significantly the batteries autonomy.

Kamdar et al. observe that improper device placement on the wrist is also a potential limitation of smartwatches for some types of sensors, for instance, heart rate can not be collected when the sensor is not in direct contact with the skin and even with skin contact variations in heart beat are observed [141]. Ahanathapillai et al. report data collection difficulties related to improper device placement when monitoring activity of elder adults[19].

Like any other wrist-worn activity device, smartwatches sometimes over-report or under-report activity levels depending on the physical characteristic of the wearer and the type of activity the wearer is doing. For instance, activities that require high levels of wrist action, such as washing hands, result in detection of increased activity level. On the contrary a wearer with limited arm movement may see an under-estimation.

Other IoT Fitness Devices

IoT fitness trackers ecosystem includes also some other devices that have not been mentioned in Section 2.4.1 and Section 2.4.1 which are the followings:

- *Smart scales*: electronic weight scales that measure both body weight and body fat mass and upload data wirelessly using Wi-Fi connection. Research has shown that daily self-weighing using smart scales can be effective for producing clinically meaningful weight loss [235].
- *Blood Pressure Monitors*: IoT blood pressure (abbreviated BP) monitor devices are normally composed by BP apparatus body and a communication module [135].
- *Glucose Level Monitors*: blood glucose monitoring reveals individual patterns of blood glucose changes and helps diabetic patients in the planning of meals, activities, and medication times [135]. IoT noninvasive glucose sensing solutions have been proposed in [136] and [254].
- *Oxygen Saturation Monitors*: pulse oximetry is suitable for the non-invasive uninterrupted monitoring of blood oxygen saturation. Sometimes smartwatches have this sensor integrated.
- *Body Temperature Monitors*: body temperature is one of the vital signs of a person and plays an essential role in the maintenance of homeostasis [215]. A temperature IoT measurement system is described in [140].

2.4.2 Smartphones and Health Mobile APPs

Nowadays over a billion people own a smartphone and over 40,000 medical apps have been deployed on apps marketplaces [112]. Smartphones assume a very important role in patient education, disease self-management, and remote monitoring of patients [179].

Smartphones combine mobile communication and computation into a single handheld-sised device. Modern smartphone devices available today in the market are equipped with multi-core CPUs and GPUs, megapixel cameras and high-accuracy built-in sensors such as GPS, accelerometers, gyroscopes, high resolution cameras, microphones, light s, magnetic field sensor, orientation sensors, atmospheric pressure sensors and proximity sensors.

In addition to calling and messaging features, smartphones are being used as an alternative to specialised sensors in medical devices. General purpose sensors on smartphones can detect various physiological signs of the users and can be exploited to diagnose a wide variety of medical conditions such as cough detection, irregular heartbeat detection, and lung function analysis. Smartphones can be exploited as well to perform fitness tracking task such as step counting using built-in accelerometer [53] [183][245] distance travelled, and calories burned.

Lee, Jinseok, et al. applied a technique called photoplethysmography (abbreviated PPG) to detect the heart rate from a fingertip using the built-in camera of a smartphone [154]. The same technique has been used to detect the heart rate from a recorded video stream of the patient's face [197]. Larson et al. realised a smartphone-based spirometer using the built-in microphone[153]. The user breathes near the smartphone's microphone and the sounds produced are processed by the software. Nan-Chen et al. developed a mobile smartphone-based system to detect and records nasal conditions (such as sneezing and runny nose) that occurs in everyday settings using audio from the smartphone's microphone [65]. Smartphone's camera has also been used to implement a medical app to diagnose melanoma [253].

Agu et al. highlight some benefits and challenges which derive from using smartphones as medical devices [18]; benefits of smartphones as medical devices:

- *Ubiquitous Deployment*: smartphones mobile app markets are available and accessible worldwide to billion of users which allows low costs of distribution for medical apps developers.

- *Ubiquitous Availability to Users*: users carry their mobile phone for a long period of the day. Research has shown that smartphones are in the same room as their owners for over 90 percent of the time [85].

- *Leveraging new hardware*: smartphone hardware is almost yearly updated to enhanced configurations. Medical apps, exploiting the rapid growth of computational capabilities, can run faster and better with few modifications.

2.5 IoT and Home Automation

The term home automation refers to the automation of the household activities including the control of lighting, heating and air conditioning, appliances, and other systems, in order to improve energy efficiency, security and the living comfort of the tenants [177].

IoT technology has profoundly changed the home automation landscape. Previous home automation systems based on wired technology that imposes constraints such as setting up the cabling, costly installation, and poor scalability are now superseded by wireless sensor networks that require less installation costs and support greater versatility [28].

The benefits that can be obtained from the use of IoT technology in home automation are manifolds [20]. Energy management is considered one of the most important application of IoT in smart homes [20]. IoT devices can manage resources more efficiently and thus reduce wasting. For example, smart lighting systems automate the action of turning the lights off when residents vacate rooms or leave their homes [155]. Smart plugs can be set to automatically switch off standby devices at night to further reduce power consumption [100]. Indoor and outdoor temperatures can be constantly monitored to better adjust settings of air conditioning systems in order to provide comfortable environments with the lowest energy consumption, without human intervention [63].

Other benefits introduced by IoT technology in home automation are related to healthcare. IoT technology can enhance home care for people with disabilities and the elderly [67]. For example, smart home automation can facilitate remote health monitoring and provide immediate clinical health care and improve access to medical services (normally unavailable in non-smart homes) [201]. Moreover, IoT based home automation systems provide entertainment and comfort to tenants, and ensure home safety and security [201].

2.5.1 IoT Domotic Devices

Two essential components of every IoT home automation system are the sensors needed to detect the human activity within a building and gauge the environmental factors, and the actuators to manipulate the physical environment. A non-exhaustive list of the most common and important IoT devices used in home automation is provided below along with a brief description of their main function:

- *Load Monitors* are used to gauge the energy consumption of a generic device. Load monitors are usually connected between the power outlet and the device to be tracked. Besides providing real-time energy consumption information, more sophisticated models can also calculate the cost associated with that consumption.

- *Smart Plugs* are remotely controlled switches and as load monitors they are connected between the power outlet and the appliance to be turned on/off. Smart plugs can be use to turn non-smart appliances into smart ones. Optionally, they can also provide feedback on the energy consumption of the appliance.

- *Smart Thermometers* are used to sense the environmental temperature and are mainly employed to realise thermostat functionalities for controlling heating/cooling systems.

- *Presence and Motion Detection Sensors* detect thermal radiation in their surrounding caused by a person approaching the presence detector. These binary sensors are employed to detect the occupancy of a room.

- *Smart Lights* are lighting devices that can be switched on/off remotely. Additionally, they allow to change the light colour and adjust the illumination level. Some smart light devices may include embedded presence detection sensors.
- *Smart Door Switches* are wireless binary sensors that are usually mounted on door and window frames to detect whether these are open or closed.
- *CO2 Sensors* or Carbon Monoxide sensors are used in indoor environments to warn home tenants if the level of CO2 concentration is dangerous.

Modern IoT smart homes are complex systems made up of an interconnected set of heterogeneous devices, as the ones described above, that assist tenants in their home activities and acquire a huge amount of data generated in the home [259].

2.5.2 Challenges in IoT Home Automation

The interaction and the information exchange among the various IoT domotic devices is essential to realise complex automation systems. However, as in the IoT healthcare field, interoperability is one of the most critical challenge in the IoT home automation due the amount of different communication standards and protocols that are being used for devices to communicate among themselves [222, 219].

The syntactic and semantic interoperability issues discussed in Section 2.2 affect IoT domotic devices as well [240]. Fortunately, a lot of progress have been made in achieving semantic interoperability through SW technology. For example, the European Telecommunication Standardisation Institute (ETSI) in close collaboration with the smart appliances industry developed The Smart Applications REFerence ontology (SAREF) which is a a reference ontology for IoT applications including domotic and building automation systems [81].

Another challenge in IoT home automation is the analysis of the data collected by domotic devices. At a very basic level, in smart homes, IoT systems assist home tenants to automatically switch on or off home attributes such as lighting and appliances. However, artificial intelligence (AI) such as machine learning techniques and automated reasoning can be employed to build more sophisticated building automation mechanisms and smart home assistive systems [222, 241]. For example, a domotic system integrated with a smartwatch could automatically alert the emergency services and unlock the main door to let them enter and rescue the injured inhabitant in case it detects a fall.

Part II

Contribution

Chapter 3

IoT Fitness Ontology

IoT health wearable devices such as smart watches, fitness bands, and wellness appliances continuously collect and store huge amounts of human physiological parameters. These data can be potentially exploited by the research community in order to gain valuable insights into people's health. However, IoT self-tracked health data come from a variety of different heterogeneous sources and in proprietary formats, which often lead them to remain confined into separate data silos. Thus, when it comes to analysing IoT health and fitness datasets, data collection and data integration have to be done manually by domain experts. This time consuming and prone to error process significantly hampers an efficient exploitation of the information available. Semantic Web technologies can be a viable and comprehensive solution for describing, integrating and sharing heterogeneous IoT datasets. The aim of this work is the standardisation of data collection and integration to permit users to get a common view of the available information. For this purpose the IoT Fitness Ontology has been designed and Semantic Web technologies have been leveraged in order to make IoT health and fitness datasets freely available to the community in a shared, semantically meaningful, and reusable manner.

3.1 Introduction

In recent years, the Internet of Things (IoT) industry has seen a proliferation of consumer devices for the purpose of health and fitness tracking. The wearable technologies market alone is anticipated to grow from 325 million connected devices in 2016 to 929 million devices by the next 5 years [11].

IoT health and fitness technologies basically encompass wearable devices such as smart rings, wristbands and smartwatches as well as non-wearable appliances like digital thermometers, scales and blood pressure meters; even health mobile apps can be viewed as smart devices or objects constituting a core part of the IoT healthcare landscape [135]. Nowadays, a wide range of individuals are able to continuously capture and digitally store a variety of personal health data (PHD).

IoT fitness devices typically exploit the embedded sensors (i.e., accelerometers, gyroscopes, altimeters) to track the physical activities of the wearer, including steps taken, stairs climbed, distance travelled and sleep hours. More sophisticated models, thanks to the advances in sensing technologies, are also capable of observing and recording multiple type of health data, including heart rate, body temperature, blood glucose level, blood pressure and other basic physiological parameters. The collected data are then augmented by secondary components providing date, time, and potentially more sophisticated outputs such as location (i.e., GPS coordinates) and user-context information.

Potentially, IoT wellness devices could give users direct access to personal analytics that can contribute to improve their health, facilitate preventive care, and aid in the management of ongoing illness. For instance, this kind of typically under-exploited information may be fundamentals for cancer prevention and the management of general oncology problems [193]. Health and fitness data collected by smart devices can provide better insights of everyday behaviour and lifestyle, and can fill in gaps in more traditional clinical data collection systems [44, 162]. Moreover, there seems to be an increasing willingness for individuals to share their PHD with others [195]. For instance, the Quantified Self (QS) movement is a notable example of this trend [214].

The sharing of the enormous amount of self-tracked health information, daily collected by users, is an important opportunity for researchers to develop novel methods to deliver high quality care and boost innovative projects which could have the potential to revolutionise the healthcare sector [44]. For instance, IoT wearable datasets have already been successfully used to study the relationship between sleep and mood problems [182], to detect unhealthy eating and exercise levels [161], and to predict human behaviour and cancer incidence [225].

However, most of the time IoT data come from a variety of different heterogeneous sources and are represented with their own proprietary format depending on the device's manufacturer. This diversity and variety which characterises the IoT health and fitness datasets along with their huge volume make sharing and integration more difficult. Therefore, today data heterogeneity is still an open issue challenge that needs to be addressed to fully exploit the potential of the IoT data.

Semantic Web (SW) technologies, based on common standards, offer opportunities to cope with the semantic data heterogeneity that hampers the integration and distribution of datasets drawn from diverse sources sharing the same context [160] [244][224]. For instance, ontologies, which constitute a formal conceptualisation of a specific domain, provide a common terminology that gives data a well-defined meaning that allow interconnection and reuse in ways other than as originally implemented or intended, independently of the type and the format of the data. Moreover, SW technologies are not only useful for converting scattered health data into valuable aggregated information, but also for sharing them.

Linked Data (LD) [48] is a set of best practises for publishing and exposing data as resources on the Web and interlinking them with semantically related datasets using SW technologies. Basically, LD is offering a worthwhile alternative to the isolated and heterogeneous data silos which dominate the IoT landscape since are based on standardised formats and interfaces. Strictly related to the concept of LD, Linked Open Data (LOD) is a term which refers to the application of the LD principles to Open Data (i.e., data that can be freely used and distributed); a classic example of LOD collection is DBpedia [13].

The next Sections present the design and development of a LOD portal which aims to become a reference point for collecting, publishing and sharing IoT health and fitness datasets in structured format. Thanks to platform, IoT data can be accessed and reused by domain experts, scientists and the web community with no restrictions by any form of licensing or patent.

Unlike other portals for sharing PHD, such as Open Humans [6] or Kaggle [4] which collect and redistribute users' raw data (i.e., data serialised in unstructured and semi-structured formats), a novel aspect of this work consists in providing a semantic representation of the IoT datasets. The domain ontology were specifically designed for IoT health and fitness devices addresses the problem of data provided in heterogeneous formats by clarifying what the data describe, thus facilitating the integration, exploration and the analysis of the datasets and promoting innovative ways to use the data.

3.2 Related Works

The related works mentioned in this section converge from different fields: (i) the relation between the market for mobile health and data inspired the work on data level; (ii) the relation between information and ontologies inspired the work on conceptual level; (iii) the paradigm of semantic-aware data integration systems motivates some of the use cases.

The market for mobile health has been growing steadily over the last years and continues to do so. The ecosystem of mobile health solution is very complex and the need to provide an integrated and open access is strong. The mHealth Developer Economics is a global research program analysing the digital health and mobile health market since 2010. The last mHealth Developer Economics survey cycle [124] describes this developing. Surveys of participants have been collected globally, with most answers coming from Europe (47%) and from North America (36%). The rest of the participants is from Asia-Pacific (11%), South America (4%) and Africa (2%). This year there are 325,000 health and fitness and medical apps available on all major app stores. Since last year, 78,000 new health apps have been added to major app stores. However, only 27% of all mobile health app publishers are already opening their apps for others immediately offering access to a wealth of valuable health data, through an API. The study reports 42% of mobile health apps connect to sensors and wearables. Overall, Fitbit is the most connected-to sensor/wearable (52%), followed by iHealth and Withings. Smart watches are becoming more attractive for mobile health app publishers, replacing other wearables. There were more than 50 Wear OS watches available in Q2/2018 from a range of third party manufacturers like LG, Fossil, Ticwatch, Asus, or Huawei. In this fragmented situation, it becomes important to furnish an integrated approach. API aggregation services are bringing together APIs from different sources into one single hub, pulling data from different sources, combining it and making it available for third parties.

Apple Healthkit is by far the most popular service with two thirds (63%) of API users opting for Apple. Number two is Google Fit (45%). All other API service providers are used by 20% or less: Open mHealth, Samsung Health, Human API, Validic and Qualcomm Life. HealthKit, as other Apple products, is restricted to run on iOS devices.

Google has a strict policy regarding what data developers can share via Google Fit. Google's policy is that health data cannot be published. Samsung Health app developers can use an existing set of data types, and can extend this set with own data types. But one disadvantage of this approach is vendor lock-in, which means further technology-driven innovations become difficult due to the vendor-specific interconnections among the different parts of the architecture [95].

The Linked Open Health-related Fitness Data system (LOHFD) was developed as an open source platform for integrating health-related fitness data that provides a unifying model to promote open data sharing and analysis using standard scalable semantic web technology stacks. For the design process, it has been followed the classical data integration paradigm that requires the creation of a common schema that consolidates schemata of the several integrated data sources, and mappings that define how they are related [123]. The global schema is the IFO ontology. LOHFD mappings each ontological term to a set of queries over the underlying data, thus reducing implementation costs and support semantic representation on a large variety of IoT data. The main benefit of LOHFD is that the combination of ontologies and mappings allows to hide the technical details of how the data are represented and stored in data sources, and to show just what the data are about.

There are multiple efforts underway that are making progresses toward addressing the challenges of open linking health-related fitness data to permit users and health professionals to get an integrated view of the available information [108].

Open Humans [6] is an open source project which aims to make more health-related data available for scientists. The online portal allows users to upload, store and share their personal data such as genetic, social-media, activities and health data gathered through IoT devices. Open Humans is a coproduced model of data community, where users are increasingly encouraged and facilitated to improve their healthcare information dataset for clinical and research purposes.

Another well-known example of data community is Kaggle [4]; it provides a platform to store and share a variety of dataset formats. Kaggle's core idea is to facilitate the analysis of data by allowing outsiders to model it. To do that, the company organizes competitions in which anyone can battle it out, from analyzing sentiments to evoke by movie reviews and how this affects the audience reaction for ranking international chess players. Kaggle adopts a crowd-sourcing approach to collect datasets from companies, scientists and users, including IoT wearable data. The dataset repository listing can be surfed by size, file type, most votes and tags.

A different solution to store and share health datasets is PhysioBank [7]. The project collects databases of physiologic signals and offers free access to the research community since 1980 via web. Successfully read and manipulate the databases require specialized software: the distributed toolkits supply methods for reading and writing signals and annotations in many formats and can be linked to user-written applications in C, C++, and Fortran.

DataGraft project [213] provides a set of tools and methodologies for open-data transformation and hosting services. DataGraft is designed to be scalable and reliable in a cloud-based environment. DataGraft's features include Resource Description Framework (RDF) data publication and querying. It was developed to provide easy-to-use tools for users who consider too costly and/or technically complex the existing approaches to data transformation, hosting, and access.

ResearchKit [9] is an open-source framework introduced by Apple that allows the use of health data directly from users' smartphones. ResearchKit collects medically relevant data obtained using the built-in capacity of the mobile device and secure data in a central repository, in compliance with regulatory requirements. The mobile application can communicate with connected devices to collect data via additional sensors, such as a heart-rate monitor on a watch or fitness band.

All these platforms are important initiatives to publish and share IoT health datasets online, but richer semantics of data are needed to resolve the heterogeneity problem and allow information integration and reuse. A notable example is Bio2RDF [35]. Bio2RDF addresses the data integration problem integrating publicly available databases in bioinformatics. Bio2RDF uses semantic web technologies to create a knowledge space of RDF documents linked together and sharing a common ontology. Bio2RDF scripts convert heterogeneously formatted data into RDF common format, without attempt to marshal data into a single global schema, Bio2RDF currently provides the largest network of Linked Data for the Life Sciences. BioPortal [218] is an open repository of biomedical ontologies that allows multiple mechanisms for content updates, provides access via Web services and provides support to integrate data from a variety of biomedical resources. BioPortal users can browse, search and visualise ontologies. The Web interface support evaluation and evolution of ontology content by providing features to add notes to ontology terms, mappings between terms and ontology reviews.

While all these projects share the common elements of longitudinal integration of heterogeneous relevant data, in some case even in health-related fitness data, each focuses on a relatively narrow set of measurements or relies on commercial or custom data storage and analysis architectures that do not provide a unifying model to promote open data sharing and analysis from multiple sources.

3.3 Web of Data

The term LD refers to a set of best practises for sharing and interlinking structured data and knowledge on the Internet by using standard web technologies [48]. The primary goal of the LD initiative is to make the Web not only useful for publishing documents, but also for sharing and interlinking single pieces of data. The movement is driven by the idea that the SW technologies facilitating the data sharing, integration, and analysis on a global scale could revolutionise the way users manage knowledge just like the Web revolutionised information sharing and communication over the last two decades.

Technologically, the core idea of LD is to use the Internationalised Resource Identifiers (IRIs) [212] to univocally identify arbitrary entities and concepts. Information about entities referred by IRIs can be simply retrieved by dereferencing the IRI over the HTTP protocol. Data about entities and concepts are then represented through the RDF language [211]. Precisely, RDF is a standardised data model which uses graphs to represent information and facts by means of triples in the form subject, predicate, object. Whenever a web client resolves an IRI associated to a triple's subject of a resource, the corresponding web server provides an RDF description of the identified entity, RDF descriptions can contain links to other RDF graphs in the triple's object. Whenever an application resolves a predicate IRI, the corresponding server responds with an RDF Schema (RDFS) [80] or a Web Ontology Language (OWL) [115] definition of the link type, that is a vocabulary or an ontology. Ontologies are a key aspect of the SW since they enable interoperability among different systems by providing an agreed-upon terminology such as the basic terms and relations in a domain of interest, and as well as rules how to combine these terms. Because the Web of Data is based on standards for the identification, retrieval, and representation of information and knowledge, and scattered entities are interconnected by links, it is possible to crawl the entire data space, fuse data from different sources, and provide expressive query capabilities over aggregated data, similarly to how a local database is queried today. For this purpose, the Simple Protocol and RDF Query Language (SPARQL) [116] is the standard language for querying, combining and consuming structured data in a similar way SQL does this by accessing tables in relational databases.

Nowadays, the great majority of IoT health and fitness datasets are accessible only in human-readable formats such as HTML pages or property data formats, therefore users must have proprietary software to access the data. Since LD is exclusively based on open web standards, data consumers and domain experts can use generic tools to access, analyse, and visualise data. Moreover, LD make use of ontologies to formally define the meaning of entities and resource so that they do not limit the ability of machines to process data automatically.

3.4 Methods

3.4.1 The IFO Ontology

In order to share the heterogeneous IoT datasets following the principles of LD, a unified representation for the different concepts as well as a formal semantics representation to these data is required. Ontologies provide a common understanding of the domain knowledge and increase data interoperability among different applications. Basically, they allow to formally describe the semantics of terms and items of the IoT fitness datasets and give explicit meaning to the provided information. Besides sharing a common understanding, another relevant benefit of a formal semantics is the support for machine-readability, which facilitates information integration and enables the application of intelligent approaches such as semantic search and automated reasoning.

Existing medical ontologies such as UMLS and SNOMED-CT [230] or IoT general sensors data annotation ontologies such as IoT-Lite [39] are insufficient to cover all the concepts and terms of the variety of IoT health and fitness datasets. Very little research has been done to provide the community a comprehensive and unified ontology specifically designed for the IoT wellness devices with the only exception of the Medical Lifelog Ontology (MELLO) [143].

MELLO is a domain ontology for representing health-related and lifelogging data including definitions, synonyms, and semantic relationships. The unified representation of lifelog terms can help to describe an individual's lifestyle and environmental factors, which can be included with user-generated data for clinical research and thereby enhance data integration and sharing. However, at the present MELLO suffers from several limitations including the lack of support of formalism such as description logic (DL) for enabling inference and knowledge integration even though the authors planned to address this limitation as a future work.

In the literature, there are several research projects, involving IoT health and fitness devices, that have explored possible methods for representing the collected data through OWL ontologies (e.g., [210, 190, 243]), however these ontologies have not been published, thus re-utilising them in other projects is impossible.

Due to the lack of a comprehensive domain ontology publicly available and suitable to be employed in a context of SW (i.e., adopting OWL as formalism) the IoT Fitness Ontology (IFO) has been designed [207, 206]. The IFO ontology is a domain ontology which aims to represent the most common and important concepts within the domain of the IoT fitness devices and wellness appliances.

3.4.2 Ontology Design Process

A reference ontology modelling process still has to be defined in knowledge engineering, however the various methodologies proposed in the literature such as the On-To-Knowledge (OTK) [234], METHONTOLOGY [101], United Process for Ontologies (UPON) [83] are slightly different from each other and encapsulate many common features, for example, being necessary an iterative processes.

To develop the IFO ontology the methodology of Ontology Development 101 [186] was followed. This methodology was chosen because it is a complete development process for building domain ontologies and it covers all the aspects needed for this study. The Ontology Development 101 process basically consists of the following iterative steps: determination of the scope of the ontology and the reference domain; reuse of existing ontologies; enumeration of important terms, definition of the classes and the class hierarchy; definition of the properties; and creation of instances.

The characteristics and functionality provided by several IoT wearable devices and wellness appliances, as well as health mobile applications available in the market, were considered and carefully analysed in order to identify the concepts described in the IFO ontology. The list of products and vendors that were taken in consideration during the design process includes: Apple Health, Microsoft HealthVault, Google Fit, Fitbit, Jawbone, Strava, Runtastic, iHealth and Nokia Health.

The key terms used in the ontology are the nouns describing generic types of physical activities and physiological parameters with no relation to specific brands. Examples of terms used about physical activities are: *Steps, Running, Walking, Swimming, ActivityIntensity, FlightsClimbed.* Examples of terms used about physiological parameters are: *HeartRate, BodyTemperature, BodyWeight, BloodPressure, CaloriesBruned.* Examples of other general terms are: *Meditation, TemporalRelationship, BodyPosture, Measure, Statistics, TimeFrame, MassUnit.*

Within the IFO ontology, classes are organised to represent the concepts in a classic hierarchical fashion. According to Uschold and Gruninger [249] there are several possible approaches in developing a class hierarchy such as: top-down, bottom-up or a combination of these two. For the IFO ontology the top-down approach was mainly used.

The ontology is build around the root class *Episode* which represent the set of the all possible events that can be measured by an IoT wellness device (a detailed explanation of the episode abstraction is given in Section 3.4.3). Object properties have been defined to model the relationships among concepts. The two most important object properties relate an episode to its measure (i.e., *hasMeasure*) and to its time reference (*hasTimeFrame*). Units of measurement were modelled as OWL individuals since are concepts that cannot be specialised anymore in the hierarchy.

To achieve a better integration with other systems and better specify the meaning of each class, references to other standardised ontologies such as SNOMED-CT were made. Personal information (e.g., date of birth) was based on FOAF ontology and the Basic geo (WGS84 lat/long) vocabulary was used for the geospatial locations.

The ontology was written using the OWL language and modelled it with Protegé as ontology-editing environment. Protegé was chosen since it is the most widely used open source ontology editor available to the OWL community and it enables the creation and representation of ontologies in OWL using a visual editor and in addition it supports automated reasoning tasks such as consistency checking and automatic classification of classes using description logic expressions [187].

The complete version of the IFO ontology is publicly available at http://purl.org/ifo.

3.4.3 Ontology Structure

The IFO ontology is built around the concept of *Episode*. An episode represents the set of the all possible events that can be measured by the IoT devices and wellness systems. For example, an episode could be the heart rate measured during a running training session by a wearable wrist worn heart rate monitor or a person's body weight measured by a smart scale.

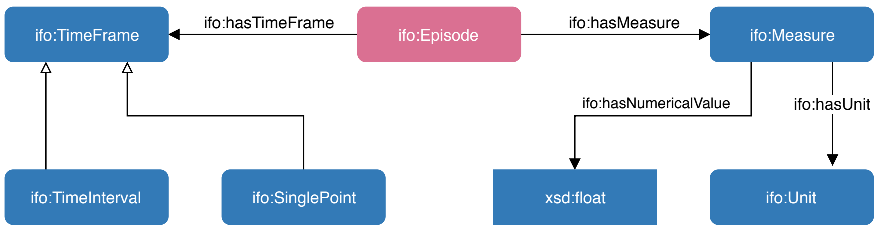

Figure 3.1: Excerpt of the IFO ontology. An Episode is any event that can be recorded by an IoT fitness device and it constitutes the fundamental abstraction mechanism of the IFO ontology. Episodes are always precisely collocated in time and can be numerical quantified.

To each episode is always associated a time reference and a numeric measurement value with the related unit of measurement. The time reference can be a single point in time or a time interval, that is, the start time and the end time of the event. These information are essential because they allow to numerical quantify the object of the event and give it a temporal collocation and duration (Figure 3.1).

Within the IFO ontology, the concept of episode is modelled by the OWL class *Episode*, which is also the root class of the entire episodes hierarchy, all the other concepts inherit the properties associated with it. The IFO ontology organises episodes in a hierarchical structure based on single inheritance. Along the hierarchy two main categories of episodes can be distinguished: (1) the physical activities; (2) the body measurements (Figure 3.2). Physical activities encompass any kind of activity involving body movement such as walking, running, swimming or steps taken. Body measurements, on the other hand, are relative to the physiological parameters of a person such as the body weight or body height or the person's vital signs such as the heart rate or the blood pressure.

Other minor categories of episodes that the IFO ontology defines, concern the sleep and the meditation.

It is noteworthy to underline that some measurements require more than a single numerical value such as the blood pressure. The blood pressure is

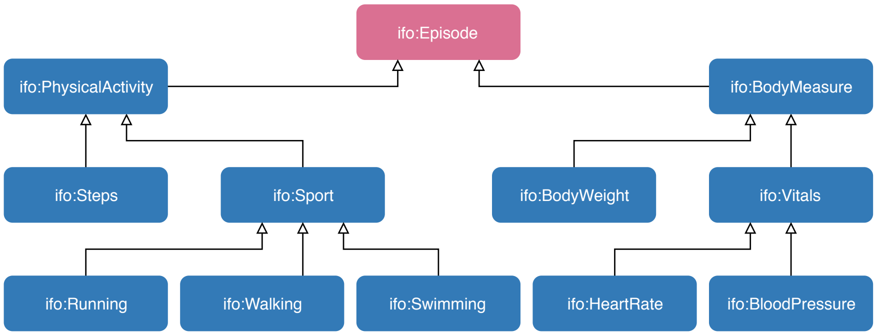

Figure 3.2: Excerpt of IFO ontology hierarchy. Episodes are grouped into two main categories: physical activities and body measurements. Physical activities are the kind of events which involve a body movement (e.g., a walk) and are typically measured by wearable devices. Body measurements regard the physiological readings normally collected using health appliances (e.g., smart scales, digital blood pressure meters).

measured in millimetres of mercury (mmHg) and is written as two numbers (e.g., 120/80mmHg). The first (120 in the example aforementioned) number is the systolic blood pressure and the second number (80) is the diastolic blood pressure. Systolic blood pressure and diastolic blood pressure according to the IFO ontology are two separated episodes.

Other fundamentals components of the IFO ontology are the OWL class *Measure* and the class *TimeFrame* which they respectively model the measurement and the time reference; these two classes are associate to the *Episode* class through the OWL properties *hasMeasurement* and *hasTime Frame* as shown in Figure 3.1. Additional metadata such as geolocation coordinates or individual's information can be optionally added to episodes (Figure 3.3).

Devices used to acquire data about an episode are represented in the IFO ontology by the class *InputSource* and are classified in *Wearable* for wearable devices, *Appliance* generic systems, *Smartphone* for mobile applications and *UserTyped* for episodes recorded manually by the user.

3.4.4 Ontology Evaluation

The IFO ontology is mostly a rearrangement and reorganisation of concepts taken from other ontologies as mentioned in Section 3.4.2 and Section 3.4.3. The abstraction mechanism of Episode allows us to describe low level concepts such as a single device sensor measurement (e.g., 36°C body temperature) as well as high level concepts like *"20 minutes of hard run followed by 5 minutes of a moderate walk"* within the same ontology.

At present there is no single best or preferred approach to ontology evaluation within the SW community, several ideas have been proposed in the literature [54]. Automatic evaluation tools constitute a valuable means for checking the technical quality of an ontology against a frame of reference and at the same time they ease the ontology diagnosis process by reducing the human intervention and costs. However, since ontologies are fairly complex systems, not every important aspects, especially those ones domain-specific related (e.g, clarity and completeness), can be automatically assessed. Therefore, human expertise is still required for a reliable validation. A dual approach based on both automatic tools and manually reviewing by domain experts was adopted in order to have the IFO ontology validated.

Automatic Tools Evaluation

During every stage of the modelling process, the IFO ontology has been continuously validated using using the reasoner HermiT [223] to check for logical inconsistencies. After completing all the modelling stages, the final form of the IFO ontology was checked and diagnosed using OOPS! [198]. OOPS! is a tool for diagnosing problems in OWL ontologies that could lead to modelling errors. The tool operates independently of any ontology development platform and is available on-line as a web application. OOPS! relies on a catalogue of pitfalls and modelling errors according to Structural, Functional and Usability-Profiling dimensions. The catalogue which was used to evaluate the IFO ontology consisted of a list of 40 pitfalls resulting of an empirical analysis carried out on 693 existing ontologies [198]. Pitfalls are also classified into three categories according to their importance level: critical, important and minor.

The results of the automatic inspection are 3 minor pitfalls and no critical or important pitfalls. More specifically the tool reported the pitfalls: P08, P13 and P36 (according to the nomenclature provided in [198]).

Pitfall P8 refers to missing annotations, which means that there are some ontology elements failing to provide a human readable annotations attached to them. However, annotating was deliberately avoided due to time constraints, some minor elements of the IFO ontology when further human readable explanation was not strictly required (e.g., *ifo:hasUnit* object property). Some other lacking labels were erroneously undetected by the tool when actually they were present. Pitfall P13 is about missing domain and range in properties. Three object properties were reported not having the domain explicitly declared and have been fixed consequently. Pitfall P36 is reported when the file extension is included in the ontology URI. In this study, pitfall P36 this was due to the modality the ontology was uploaded to the scanner. This problem has been fixed in the released version of the IFO ontology.

Overall, there is a high level of correctness according to the results provided by the tools that were used to perform the diagnosis of the IFO ontology.

Evaluation of Domain Experts

Two domain experts, with a background in biomedical engineering and knowledge engineering, who did not participate in the modelling process validated the IFO ontology according to some of the metrics described in [119, 118, 102, 103, 188]. In particular, since a quantitative assessment using the evaluation tools had already been obtained, experts were requested to focus on the evaluation of the following qualitative parameters: Accuracy, Clarity, Completeness, Adaptability and Conciseness. The evaluators were provided with the IFO ontology in OWL format and a complete schema of the ontology similar to the ones shown in Figure 3.2 and Figure 3.1. Both the domain experts independently examined the material and return us a feedback for each evaluation parameters.

Both the experts agreed on the correctness of the IFO ontology in modelling the concepts of the reference domain, the overall Accuracy has been unanimously evaluated as good. Labels and concept description have been taken into account as well during the accuracy evaluation.

Also the Clarity has been valuated positively, the experts took into consideration also the documentation provided along with the OWL format. The abstraction of the Episode due to its simplicity can be quickly understood by ontology users and additionally concept labels are immediately recognisable and often accompanied with exhaustive descriptions.

The aim of the IFO ontology is to model the most common and the most important concepts of the IoT fitness devices and not also the minor concepts that can be found in the domain, for example, a concept which is related only to a specific vendor of a specific device is out of the scope of the IFO ontology. Given this premise, the experts agreed that the IFO ontology reaches a satisfying degree of Completeness.

About the Adaptability, the experts verified that the ontology can be eventually easily extended with concepts not already included in the present version. Moreover, the IFO ontology can also be monotonically extended or specialised maintaining the same fundamental abstraction mechanism of Episode. For example, the Physical Activities hierarchy can be extended to keep track also of with weight-stack gym machines training sessions. The IFO ontology has been considered highly extendable and adaptable.

Experts agreed that Conciseness has been achieved thanks to the simplicity of the mechanism of the Episode and the avoidance of redundant axioms. Moreover, all the concepts modelled have been evaluated relevant to the reference domain.

3.5 Web Portal

The aim of this study was to design and develop a LOD-based web portal in order to collect health and fitness data gathered from consumer health IoT

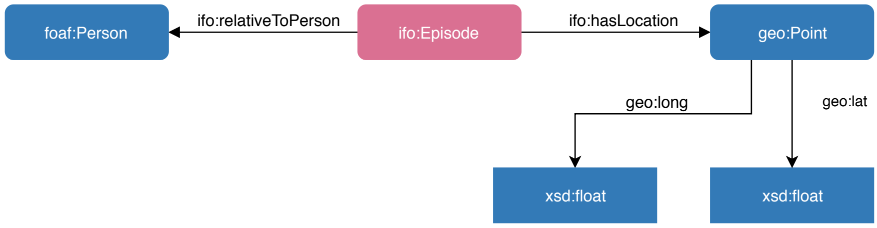

Figure 3.3: Excerpt of the IFO ontology. Episodes can be augmented with additional metadata such as individual's personal information or geolocation position.

devices, and make them freely available on the Web. For the design process of the system the detailed set of recommended practices for creating and publishing LD sources in the Health Care and Life Sciences (HCLS) domain as described in [166] were mostly followed.

The resulting platform is capable of: (1) collecting IoT fitness data manually entered by users or automatically retrieved from remote repositories; (2) integrating and storing IoT datasets semantically annotated according to a reference ontology; (3) visualising information through a customisable dashboard; (4) sharing datasets adhering to LD principles. For the development of the system a four-tier architecture was adopted. A layered architecture makes the various parts of the system independent and logically separated, and single components, replaceable and upgradeable. For instance, at the moment, data can be entered to the system manually by users or downloaded automatically from the Cloud, the layered architecture would allow us to add a third method for collecting data (e.g., directly from devices) without affecting the rest of the system.

As can be seen in Figure 3.4, the four-tier architecture consists of the following layers: (1) the data retrieving layer; (2) the data processing layer; (3) the service layer: (4) the presentation layer. The data retrieving layer collects IoT datasets from users or automatically from remote servers. The data processing layer transforms the IoT raw data in semi-structured formats into an RDF graph, datasets are thus semantically annotated accord-

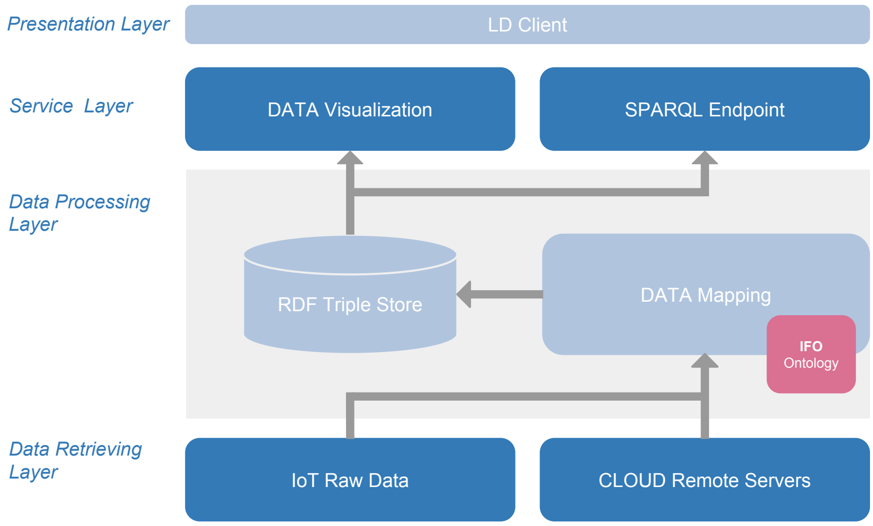

Figure 3.4: LOD system architecture with all the four layers from the data retrieving layer to the presentation layer.

ing to a reference ontology and stored within a NoSQL database (i.e., a triplestore). The service layer controls data access and bridges the clients to the system via service protocols. The presentation layer allows users to interact with the system using either the web based dashboard or the SPARQL endpoint. The server side part of the web portal was developed using the JavaServer Pages (JSP) [3] technology. The experimental web portal is available at: *http://137.204.74.19:8080/IFOPlatform/welcomePage.jsp*. On the *Help* page of the web portal a video tutorial and some sample datasets for testing purposes are provided.

3.5.1 Data Retrieving

Gathering and integrating in an homogenised way the huge volume of IoT health information is an extremely challenging task because data are spread across different platforms in heterogeneous formats (data silos). Datasets can be manually entered by users to the system or can be automatically retrieved from cloud storage systems since many vendors allow to obtain the data via their servers through public APIs.

Most IoT fitness vendors grant access to data stored on their servers using authentication mechanisms such as the Open Authorisation (OAuth) [126]. OAuth is an open standard and provides external applications secure delegated access to a server on behalf of the owner. OAuth specifies a process to authorise access to the resources without sharing the user credentials. The system allows users to automatically retrieved data from Fitbit and Nokia Health servers. Once the user has given the permission to access health data on his behalf, the system can periodically download data without further user intervention.

IoT fitness servers do not rely on a standard for exchanging data. Without exception, each vendor defines its own specific proprietary API interfaces. Server-specific software has to be written to retrieve the data once the access authorisation has been granted.

The most common serialisation formats used for IoT health and fitness data are the Extensible Markup Language (XML), the Comma Separated Value (CSV) and the JavaScript Object Notation (JSON). The data formats promoted by the Health Level Seven (HL7) standardisation group seem to be ignored and not taken into account by any IoT fitness producer.

For example, the JSON code shown in Listing 3.1 is the response obtained, after being authenticated and authorised to the Fitbit server, by executing an HTTP GET request using Fitbit proprietary APIs.


```
{
    "weight":[
    {
        "bmi":23.57,
        "date":"2015-03-05",
        "logId":1330991999000,
        "time":"23:59:59",
        "weight":73,
```

```
        "source": "API"
    },
    {
        "bmi":22.57,
        "date":"2015-03-05",
        "logId":1330991999000,
        "time":"21:10:59",
        "weight":72.5,
        "source": "Aria"
    }
    ]
}
```


Listing 3.1: An excerpt of body weight data retrieved in JSON format using Fitbit proprietary APIs.

In Listing 3.1, the output given consists of a list of all user's body weight log entries for a given day using units in the unit measurement system which corresponds to the *Accept-Language* HTTP header provided during the request. The specific device by which the data have been collected, date and time, and the numerical value of the measurement are all specified within the response.


```
Date,Weight,"Fat mass","Bone mass"
"2017-08-10 20:31:00",82.00,10.00
"2017-08-07 11:10:50",81.00
```


Listing 3.2: An excerpt of body weight data collected by a Nokia Health smart scale in CSV format.

Listing 3.2 shows body weight data collected by Nokia Health smart scale. The output is in CSV format and has been obtained using the export function provided on the Nokia Health online dashboard. In addition to the body weight the CSV file might also contain information about the fat mass, the bone mass, the muscle mass and user personal comments (not shown in Listing 3.2). It is noteworthy to highlight that the date and time are stored within a unique string while in the Fitbit example (Listing 3.1) they are separated.

```
<Record
 type="HKQuantityType
             IdentifierBodyMass"

 sourceName="Lifesum"
 sourceVersion="6.2.0.7"
 unit="lb"
 creationDate="2016-06-0816:47:26-0400"
 startDate="2016-06-0800:00:00-0400"
 endDate="2016-06-0800:00:00-0400"
 value="150"
 />
```

Listing 3.3: An excerpt of health data manually exported in XML format from the Apple Health app.

In Listing 3.3 is shown an excerpt of data manually exported from Apple Health7 in XML format. Information is about the body weight of the user again but in this serialisation also the unit of the measurement and the data source (in this case a mobile application) are included.

The excerpts Listings 3.1, 3.2, 3.3 are straightforward examples of the issues related to the heterogeneity of data representation and serialisation formats used within the IoT fitness domain. The same concept of body weight is represented in three different ways and serialised in three different formats.

Once the IoT data is retrieved manually of automatically from remote servers, a copy is maintained within an offline repository for archiving purposes.

3.5.2 Data Processing

LD recommendations require datasets to be published in RDF format. Since IoT datasets are retrieved in semi-structured formats (i.e., CSV, JSON, XML) a process to convert them to RDF is needed. During the conversion process, besides the ontology mapping, also well-formed IRIs are generated and assigned to entities within the datasets.

For the mapping process the RDF Mapping language (RML) [88] and the RML Processor for its execution were chosen.

RML is a declarative source-independent mapping language which allows to express customised mapping rules for converting heterogeneous resources into RDF graphs according to a reference ontology in an integrate and interoperable fashion. RML extends the W3C standard R2RML [8] which is defined to express customised mappings only from data stored in relational databases. RML keeps the mapping definitions as in R2RML but encompasses broader set of possible input sources. Moreover, RML provides the vocabulary for defining the iterator pattern over the input data which allows us to explicitly specify how the source data have to be accessed. Iterator patterns make use of target-specific query languages. For example, an XPath expression can be used to specify an iterator over an XML document while a JSONPath expression can be defined in a similar way for a JSON document.

These characteristics of RML make it particularly useful within the IoT fitness context because, as already stated, different vendors use different data and serialisation formats to represent and store information about the same concept. For instance, Fitbit stores data about the body weight in JSON format, Nokia Health serialises the same type of data in CSV format while Apple Health does the same in XML, as shown in Section 3.5.1.

RML mapping specifications are based on one or more *Triples Maps* which define how the triples (i.e., the resulting RDF graph) are generated. Essentially a triple map contains a rule to generate zero or more RDF triples which share the same subject for each extract of data from the input source. A single triples map is composed by the *Logical Source*, the *Subject Map* and zero or more *Predicate-Object Maps*.

As an example, Listing 3.4 shows a set of RML triples maps which can be used for generating an RDF graph starting from the Fitbit data about the body weight as proposed in Listing 3.1. The logical source consists of the reference to the input source to be mapped, in this case the *fitbitWeight.json* file. The *Reference Formulation*, pinpoint by *rml:referenceFormulation*, specifies how references to the data occurs and, since RML uses references relevant to the input source, in this case JSONPath is used. The iterator specifies how to iterate over the input data, here is specified by the JSONPath expression: *$.weight*. The subject map consists of the template that defines the URI pattern used to generate the subject of the triple and optionally its type. In this case a blank node is generated and the triple is typed as *fo:Measure*; *fo* is the name space used for the IFO ontology. A *Predicate Object Map* consists of a *Predicate Map* that specifies the predicate of the triple and an *Object Map* which specifies the object (one or more) of the triple. Specifically in this case a JSONPath expression is used to point to the body weight value in the source *rml:reference "@.weight"*. The resulting RDF graph is shown in 3.5

```
<#FitbitBodyMass>
rml:logicalSource [
 rml:source "fitbitWeight.json";
 rml:referenceFormulation ql:JSONPath;
 rml:iterator "$.weight";
];

rr:subjectMap [
 rr:termType rr:BlankNode;
 rr:class ifo:Measure;
];

rr:predicateObjectMap [
 rr:predicate ifo:hasNumericalValue;
```

```
 rr:objectMap [
  rml:reference "@.weight";
  rr:datatype xsd:float;
 ];
];
```

Listing 3.4: An example of RML triples map which can be used to generate an RDF graph starting from a JSON file about body weight data collected by a Fitbit IoT smart scale.

```
_:kWRuix2ft9 a ifo:BodyWeight ;
 ifo:hasMeasure _:fxbMJQzZG8 ;
 ifo:hasTimeInterval _:CrHFdBYBD8 .

_:fxbMJQzZG8 a ifo:Measure;
 ifo:hasNumericalValue "73"
 ^^xsd:float;
 ifo:hasUnit ifo:kg .

_:CrHFdBYBD8 a ifo:TimeInterval ;
 ifo:endDate "2015-03-05 23:59:59"
 ^^xsd:dateTime ;
 ifo:startDate "2015-03-05 23:59:59"
 ^^xsd:dateTime .
```

Listing 3.5: RDF graph representing IoT health data annotated according to the IFO ontology.

RML significantly simplifies the development of a mapping specification for the same concepts since the definition of the triple structure has to be specified only once and can be reused across other sources in the same or different formats. Moreover, RML mapping specifications can be also generated in a semi-automatic way [89].

Several mapping specifications, manually written, for different devices are already available and ready to use within the system and users can also add their own specifications. As soon as the triples are generated they are loaded to the triple store.

3.5.3 Data Sharing

IoT health and fitness datasets transformed into a well-organised LD structure are exposed, on the portal, through a public SPARQL endpoint. A SPARQL endpoint essentially enables users or software application clients to query the RDF data via the SPARQL language. From a SPARQL query perspective, once the data is represented as RDF and exposed via a SPARQL endpoint, the different storage modalities become irrelevant. However, SW systems, such as LD portals, usually build on triplestores as their main data storage. Triplestores provide data management and data access via APIs and query languages for RDF data. For the portal, Fuseki [1] was adopted as triple store and SPARQL endpoint.

3.5.4 Data Visualisation

Information visualisation is an important component of LOD portals since it increases accessibility of LD-based systems [148]. The main objective of information visualisation is transforming and presenting data into a visual representation, in such a way that users can explore and use the data.

The system allows users access their personal data through a web-based visualisation dashboard which provides multiple views of their integrated datasets.

Health data representation methods must be flexible in order to cover the needs of users with different backgrounds and requirements. RDF data model offers unique opportunities since it enables to bind data to visualisations in unforeseen and dynamic ways [59]. For instance, when an information visualisation technique requires certain data structures to be present, these data structures can be derived and generated automatically from reused vocabularies or semantic representations, in this way it is possible to realise a largely automatic visualisation workflow.

To exploit the flexibility offered by RDF the dashboard was made highly customisable by allowing expert users to define the information to be displayed on charts using a user-made SPARQL query (Figure 3.5). Federated queries are also possible within the dashboard.

Since writing a SPARQL query is a challenging task for nontechnical users, the dashboard was provided with several preset queries for visualising common information in the form of time series such as the heart rate or the blood pressure readings.

3.6 Results

The main result of this work is a system which is able to integrate data from multiple heterogeneous IoT health and fitness sources and expose them in structured format so that they can be accessed and queried in a uniform way using standard language. Data visualisation in a personalised manner is also possible through a web dashboard.

The system was tested using data about body weight, blood pressure and heart rate collected using three different IoT fitness devices. The choice of the devices and the typology of data was based on the different serialisation formats they adopt to store the collected data. Body weight data was generated and uploaded to the Nokia Health server through the dedicated smartphone app since Nokia Health give access to the datasets in CSV format. The blood pressure readings were partially uploaded to the Nokia Health server and partially stored on a smartphone within the Apple Health app, in the latter case data is stored in XML format. Heart rate information was collected through a Fitbit wristband (in JSON format) and an iOS smartphone app [168]. Data from Apple Health were manually entered to the system while body weight (Nokia Health) and heart rate readings (Fitbit) were automatically downloaded by the system from their respective remote servers. All the IoT datasets provided have been correctly uploaded and correctly transformed to structured data.

The web dashboard was used to visually explore the homogenised datasets and SPARQL for performing some statistical operations over the homogenised

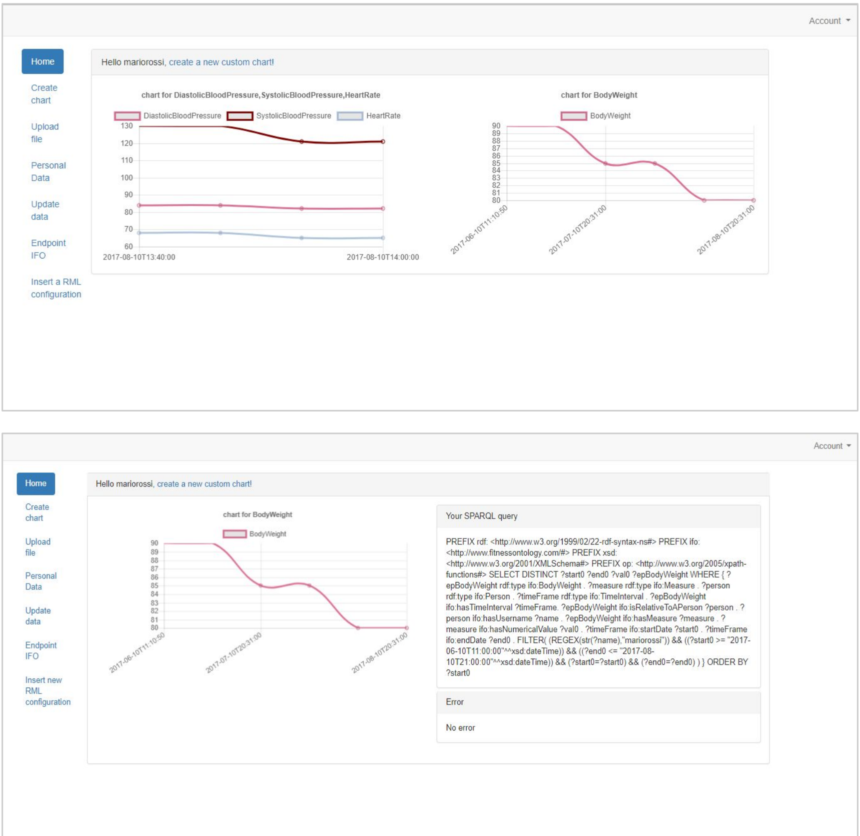

Figure 3.5: Two screen shots of the web-based dashboard showing some vital signs data charts and a customised body weight chart generated through a user defined SPARQL query.

datasets. It is noteworthy to underline that such queries have operated regardless of how the original data were stored and represented in the IoT sources (i.e., CSV, JSON and XML in this study). Moreover, experimental datasets could have analysed together with some other data from another LOD portal in the same way, since once the data is represented as RDF and exposed via a SPARQL endpoint, the different storage modalities become irrelevant from a SPARQL query perspective. A similar task without the support of SW technologies would have been not trivial to formulate and execute, a federated query "on the fly", as the one aforementioned, virtually impossible.

3.7 Discussion and Conclusions

The enormous amount of self-tracked health information collected by users through smart fitness devices, offers important opportunities to the research community. The process of gathering and integrating data from scattered IoT sources is normally done manually by researchers and domain experts. This process is not only cumbersome but also significantly time consuming and in many cases, error prone. An effective and efficient exploitation of the IoT health and fitness data requires methods for accessing, integrating, interpreting and analysing datasets from multiple distributed sources in a unified way.

It was designed and developed an LD-based web platform which is capable of collecting and retrieving IoT health and fitness datasets from users and SQ enthusiasts in order to make them freely available to the research community. The system can convert heterogeneous IoT raw data collected by a multitude of different devices into RDF graphs. The homogenised datasets are stored in structured format and exposed publicly via a SPARQL endpoint for accessing and querying. However, the study was not limited to merely convert the raw data into RDF and publish them online. The IFO ontology was also leveraged to make the IoT health and fitness datasets available in a shared, semantically meaningful, easily discoverable, and reusable manner. Additionally, the web portal was provided with data visualisation capabilities so that the platform can also be used as a personal health record (PHR) system. Users can visualise and explore their integrated datasets through a customisable web dashboard. The LOD portal presented in this Chapter is the first of its kind specifically designed to gather and share IoT health and fitness datasets. In this study, it has also been demonstrated that SW technologies can be a viable and comprehensive solution for describing and integrating the heterogeneous IoT health and fitness data. In particular it has been showed that the LD initiative may offer unprecedented opportunities for exposing the information collected by IoT devices inasmuch LD rely on structured RDF graphs that can be queried uniformly via SPARQL. LD enable the fusion of local and public data in a very powerful way, thus overcoming the main issues of data silos.

3.7.1 Limitations

There are some limitations to this work. Firstly, the sample queries and the amount of data that were used to test the system is relatively small, and a more robust and rigorous evaluation along several dimensions (e.g., performance, query results, robustness, devices supported) is recommended. Secondly, policies and practices that relate to privacy of health information should be elaborated further. Even if the system shares information only in anonymous form, when two or more sources of personal data are combined the risk of revealing a person's identity increases significantly [239, 14, 203]. Finally, although the IFO ontology covers a vast set of IoT fitness concepts, describing the datasets more robustly with domain-specific, additional ontological vocabularies and interlinking with more ontologies is recommended.

3.7.2 Future Works

For future research, additionally to address the limitations aforementioned, an integrated search engine for supporting a better data discovery and access which can potentially make the LOD portal more interesting and usable by researchers and healthcare professionals should be implemented. Moreover, the reasoning and inference mechanisms of the SW technologies and the IFO ontology should be leveraged to enhance the dashboard by providing it with more advanced data analysis capabilities specific for the IoT health and fitness datasets.

Chapter 4

Semantic Smart Home System

Despite the pervasiveness of IoT domotic devices in the home automation landscape, their potential is still quite under-exploited due to the high heterogeneity and the scarce expressivity of the most commonly adopted scenario programming paradigms. The aim of this study is to show that Semantic Web technologies constitute a viable solution to tackle, not only the interoperability issues but also the overall programming complexity of modern IoT home automation scenarios. To this purpose,a knowledge-based home automation system has been developed. Within the system, scenarios are the result of logical inferences over the IoT sensors data combined with formalised knowledge. In particular, this Chapter describes how the SWRL language can be employed to overcome the limitations of the well-known trigger-action paradigm. Moreover, it shows how through various experiments in three distinct scenarios, the feasibility of the proposed approach and its applicability in a standardised and validated context such as SAREF has been demonstrated.

4.1 Introduction

Over the last few years, the home automation sector has seen significant adoption of IoT devices performing the most various sensing and actuating capabilities for a plethora of different domotic tasks. Basically, IoT domotic devices are size-contained wireless systems which can be accessed and programmed through the internet to monitor or control home attributes such as air conditioning and heating, lighting, surveillance and home appliances.

IoT home automation, besides increasing entertainment and user comfort, can provide several potential benefits in other crucial areas such as home safety [16], energy efficiency and preservation [165], and elderly care [248]. However, current home automation systems do not fully exploit the intrinsic potential of IoT devices due to two critical challenges they face: the high heterogeneity of the devices and protocols which results in limited interoperability (a common issue that plague the IoT landscape in general); and the limited expressivity of the paradigms adopted to program domotic scenarios [12]. In fact, the majority of commercially available IoT programming environments are proprietary technologies based essentially on the low level abstraction trigger-action model which is limited to processing only a small number of different input data sources (normally only sensor data gathered by devices from the same vendor) and lacks reasoning features.

For example, a typical current domotic scenario for room temperature regulation consists of turning on the air conditioner set to the desired temperature and letting the internal thermostat do the rest. However, a smart IoT system could be embedded with the knowledge to make it be able to autonomously find the best way to cool down the room temperature. For instance, by combining data from internal thermometers and online weather services it can decide to just open the window and let the fresh air in instead of simply turning on the air conditioner thus saving energy. Additionally, the same system could automatically close the window if it starts raining outside or when the residents leave the house thus guaranteeing safety.

Achieving such complex behaviour requires the smart home system to be able to combine and analyse data coming from heterogeneous sources according to some formalised knowledge [208, 226]. Consequently, automated reasoning capabilities and the adoption of a higher abstraction programming paradigm becomes essential as well [75].

Semantic web (SW) technologies consist of a set of open recommendations for associating data to their formal meaning and have been shown to constitute an appropriate means for achieving data interoperability in IoT systems [24, 133]. Additionally, SW naturally enable inference capabilities over semantically annotated data that cannot be obtained using other traditional programming languages. Numerous studies in the literature have shown that SW technologies can be a suitable approach to tackle the complexity of many *specific tasks* in *specific areas* of modern IoT home automation [84, 199, 94].

The use of SW technologies, in particular OWL ontologies, as a means to overcome the poor interoperability in the IoT field has been recently largely investigated [202, 208, 206]. One notable example in the smart home field is the Smart Appliance REFerence ontology (SAREF) [81]. SAREF is an OWL-DL ontology, created in close interaction with the industry, that aims at formally describing the core concepts in the smart appliances domain. The ontology defines concepts for modelling the devices, their tasks and the functions they perform to accomplish the tasks. It also enables the description of the device energy profile and power profile. The ontology is designed to be easily extendable and can be used as basis for creating more specialised ontologies such as SAREF4Health, an extension for IoT-based healthcare systems [178]. Moreover, SAREF has been standardised by the European Telecommunications Standards Institute (ETSI) and experimentally validated [251].

Ruled-based programming approaches such as the trigger-action paradigm are widely employed in actual IoT home automation systems for defining domotic scenarios due to their simplicity and intuitive use [32]. For example, IFTTT is one of the most popular tool for programming IoT scenarios using trigger-action rules [2]. However, even though IFTTT partially extends the expressivity of the simple trigger-action model through the integration of web services, it suffers from several limitations such as a low-level abstraction and low generalisation due to the lack of actual semantics support and reasoning capabilities [12, 75]. Indeed, besides providing semantics to IoT data, SW technologies also provide support for dealing with the collected information, that is they naturally enable reasoning capabilities over the semantically annotated data, especially by means of rule languages [64].

In [74] Corno et al. employed SW technologies to overcome the low level abstraction of IFTTT rules in end-user development (EUD) environments. The authors created EUPont, a high-level OWL ontology that provides abstract representations for EUD programming environments for the IoT.

The applicability of SW ontologies and rule languages in home automation has been investigated by Bonino et al. [50]. In their work, the authors employed an OWL ontology (i.e., DogOnt ontology [49]) to provide a common semantics and features description for the devices involved, and two rule languages (i.e., SWRL and Jena rules) to perform reasoning. In their system, rules are defined to evaluate structural and state properties of the home environment.

Fensel et al. in [97] describe the SESAME-S project (SEmantic SmArt Metering Services for Energy Efficient Houses) which makes use of linked data to assist home tenants in making informed decisions and controlling home energy consumption. Owl ontologies are used to semantically annotate data regarding automation devices, consuming measurement and energy pricing. Rules are employed to implement policy-based decision-making mechanisms.

More recently, in [217] Saba et al. have proposed a system for energy management in smart home environments. The OWL ontology on which the system builds up provides a formal representation of the energy aspects of the appliances and other domotic environment elements such as the extent by which they positively or negatively influence the consumption of electrical energy. SWRL rules are used as reasoning mechanism to implement energy saving scenarios without compromising tenants' comfort.

Previous works in the domotics field that have adopted SW technologies focus only on specific home automation tasks (e.g., energy management). Accordingly, automated reasoning through OWL and SWRL is performed over solely *ad hoc* written ontologies.

Indeed, the necessity of an ontology-agnostic approach in SW solutions have been clearly highlighted in [220] for building-level automation systems. In this work, the authors propose BRICK (Building's Reasoning for

Intelligent Control Knowledge-based System) as an integrated system for intelligent building energy and security management.

Stemming from these results, the aim of this study is to suggest a unifying approach in home automation systems by demonstrating how the Semantic Web Rule Language (SWRL) can be employed as a *general purpose* programming paradigm to implement advanced domotic scenarios which are not limited to work only in a specific domotic area or rely only on *ad hoc* ontologies. For this purpose, the Semantic Smart Home System (SSHS), a knowledge-based system which is capable of performing home automation facilities by executing SW rules over *in situ* collected IoT sensor data potentially combined with external information sources (e.g., curated ontologies or third party web services) was developed. This work builds upon the existing standardised ETSI Smart Applications REFerence (SAREF) ontology[1], which, as mentioned above is specifically designed to enable semantic interoperability in IoT systems including domotics [81, 251]. It is important to note that SAREF is a general purpose ontology in the IoT domain, it has not been developed for a single specific project, therefore it cannot be considered an *ad hoc* solution.

The experiments that have been carried out showed the feasibility of the proposed approach and the efficiency of the SSHS in realistic real-life settings.

4.2 Materials and Methods

The main objective of this study is to demonstrate that SW technologies can be employed to realise home automation scenarios that can fully exploit the potential offered by modern IoT devices. Indeed, in order to achieve an overall higher level of automation, the heterogeneous amount of information collected by IoT devices crucially requires to be integrated and analysed accordingly to some sort of formalised knowledge, such as OWL ontologies.

To achieve this goal the SSHS which is a knowledge-based framework

[1] The SAREF ontology is available at https://saref.etsi.org/

that can execute complex home automation scenarios was developed. In SSHS, automatic actions are the result of SWRL rules executed over semantically annotated IoT data potentially combined with external information sources such as curated ontologies or web services. In SSHS, semantics also provides the expressiveness and abstractions which facilitates the task of tackling the programming complexity. That is, in SSHS, rules are expressed in terms of high level entities rather than the respective lower level of the sensor raw values or device specific actuator commands.

The SSHS operates within a SAREF-based environment. A SAREF environment, in this context, is essentially composed of two elements: a set of knowledge bases and a knowledge engine. The term knowledge base refers to a general entity which can communicate and exchange data semantically annotated according to the SAREF ontology or one of its extensions. For instance, any IoT device, such as a door switch or a thermometer, in a SAREF environment can be seen as a knowledge base. All the knowledge bases are connected to the knowledge engine through a smart connector (i.e., a generic component that is being developed by the InterConnect consortium[2]). Knowledge bases configure their smart connector with their specific capabilities and RDF graph patterns. The knowledge engine uses these to function as the coordinator system that allows the knowledge bases to exchange information to each other. The RDF graph patterns are used by the knowledge engine to route the data. Further information about the Knowledge Engine can be found in the InterConnect public git repository[3].

The SSHS can be seen as a SAREF knowledge base that once connected to the knowledge engine periodically receives sensor data gathered from the IoT devices, executes semantic rules over them, and sends commands back to control the actuators. Since the SAREF ontology provides a common semantics, rules can be written device vendor independently. In Figure 4.1 is shown an example of SAREF setup in which four IoT devices collect

[2]InterConnect consortium:
https://interconnectproject.eu/consortium/

[3]InterConnect public git repository:
https://gitlab.inesctec.pt/interconnect-public/knowledge-engine/

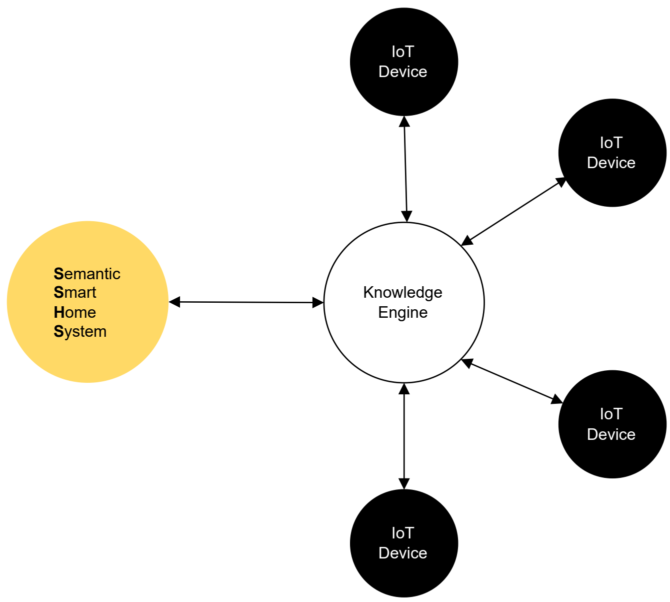

Figure 4.1: The schema depicts an example of SAREF environment setup, as intended in this study, composed of four IoT devices that exchange data in RDF format, through the knowledge engine, with the Semantic Smart Home System.

data from the environment and exchange them with the SSHS that can eventually send commands back to them.

Figure 4.2 depicts the main architecture of the SSHS. The SSHS is made up of three main components: the update module, the core system and the actuate module. These components realise three phases which are executed in order and cyclically.

The first phase is the update phase which is performed by the update module. During the update phase, the system gathers all the sensor data measured at that moment by the IoT devices installed in the house. Optionally, the incoming data can be arbitrarily augmented through a software procedure. For example, if a physical device does not provide in its out-

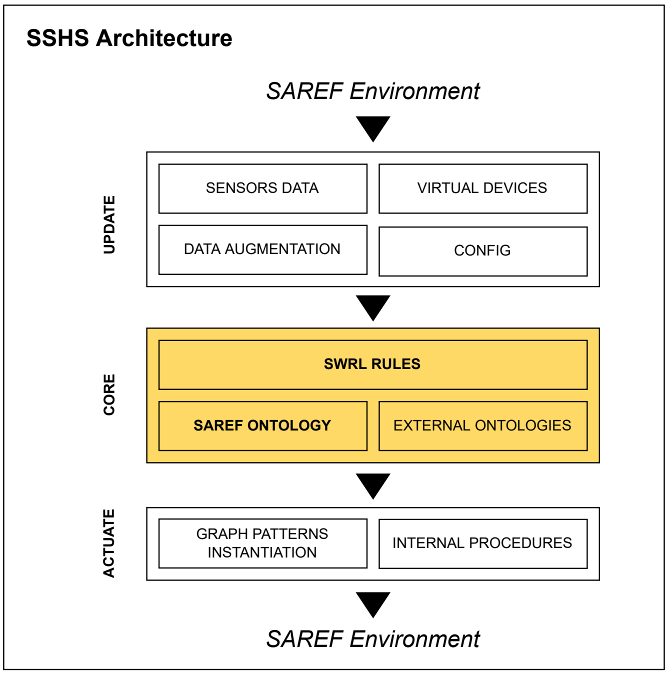

Figure 4.2: The Semantic Smart Home System architecture. The system is made up of three main components: the *update component* which collects data from the SAREF environment and constructs the RDF graph representing the current house status, the *core component* that executes the SWRL rules over the content of the main graph, and the *actuate component* which inspects the reasoning result and sends commands back to the actuators. These tasks are executed sequentially in a loop.

put a timestamp, it can be added at this stage. Data from virtual devices (i.e., web services or external data sources) and configuration information is collected as well. The output of the update phase is an RDF graph which represents the current status of the house combined with all the other information gathered.

The second phase is the rules execution phase that is performed by the core system. The core component represents the most important part of the SSHS. It executes the semantic web rules over the RDF graph produced during the previous phase. The semantic web rules are expressed using the SWRL language. The SAREF ontology and optionally other external OWL ontologies, takes part in the inference process as well. In our SSHS implementation the OWL classification and the rule execution are performed by the reasoner Pellet which is an open source OWL-DL reasoner that also features an embedded SWRL inference engine [229].

The third and the last phase is the actuate phase that is performed by the actuate module. The actuate module first analyses the result of the reasoning process and then accordingly it generates the graph pattern instantiations (i.e., the device commands) to be sent to the knowledge engine in order to operate the actuators such as turning on a lamp or closing a shutter. Along with graph pattern instantiations, during the actuate phase internal procedures are executed to modify the status of the virtual devices.

4.2.1 IoT as Data Sources

Reasoning over IoT sensor data combined with external information sources is the key feature of the SSHS. Within the SSHS every data source can be modelled as an IoT device. In SSHS an IoT device can be of three kinds: physical device, augmented device and virtual device.

Physical devices correspond to the actual IoT devices materially installed inside the house. For instance, a physical device could be a door switch sensor mounted on a door frame or a thermometer which gauges the room temperature. For each physical sensor in the house, the update module collects its status information as an RDF graph as it comes as output from the device itself.

An augmented device is basically a physical device to whose output graph one or more pieces of extra information are added through a software procedure during the update phase. This is necessary because unfortunately IoT devices do not always provide by default all the essential information that might be needed to write rules, such as their status history. For instance, most of the colour changing light bulbs available on the market provide in their status only the current colour which is actually being displayed while sometimes it could be necessary to keep track of the previous configurations to restore a previous state after temporarily changing it.

A virtual device is a generic IoT device whose functionalities are fully software emulated. Virtual devices do not have a corresponding physical device installed in the house. For example, the information that an IoT weather station provides (e.g., temperature, humidity, wind speed, rainfall and solar radiation) can be also obtained from an online weather web service thus making unnecessary (at least in some cases) the presence of an actual physical device in the house. Virtual devices can also be used to provide functionalities that are not natively available in SWRL or OWL run-time environments, such as functions to retrieve the current date. However, in SSHS, a clock can be easily implemented as a virtual IoT that provides as output the actual date time. Virtual devices can either be implemented as software procedures invoked by the update module or as separate SAREF knowledge bases connected directly to the knowledge engine.

4.2.2 Implementation details

A prototype of the SSHS has been developed using the Python language. The Owlready2 [152] framework was used to connect the update and the actuate component to the core component which is enabled by the Pellet reasoner [229]. Pellet is an open source OWL2-DL reasoner that natively supports SWRL rules execution.

Some demo code is available at https://github.com/robertoReda/sshs.

4.2.3 Rules Writing and Scenarios

A home automation scenario refers to a set of actions that are performed when certain conditions are met. Most of the current home automation tools adopt the trigger-action model as a home automation scenario programming paradigm [12]. In trigger-action systems the desired behaviour is specified by means of rules in the form "if-then" where the "if" part of the rule checks whether a particular event (i.e., the trigger) has occurred and the "then" part specifies the action that should be executed in response. For instance, a typical scenario rule could be: "If the leak sensor detects some water, then turn off the washing machine". Rule-based languages provide an intuitive way to program home automation scenarios especially when IoT devices are involved [12]. However, IoT home automation tools that do not adopt semantics suffer from several important limitations. First, the impossibility to define generic sets of rules for devices which have similar functionalities instead of vendor specific rules. Second, triggers and actions can be specified only in terms of device output values and device commands which implies a low level of abstraction. Finally, actions are determined on the sole base of data input sources due to the lack of reasoning capabilities.

The Semantic Web Rule Language (SWRL)[181, 130] allows defining rules in terms of OWL entities, that is, it combines the ease of rule-languages with the capability to perform automated reasoning. SWRL is the means by which domotic scenarios are programmed in the SSHS. In SSHS automatic actions are determined by analysing the IoT input data according to knowledge expressed in OWL ontologies or in the rules themselves. In other words, in SSHS automatic actions are the result of logical inferences drawn over the RDF graph that represents the current status of the house.

Technically, SWRL provides a high-level abstract syntax for horn-like rules fully compliant with OWL semantics. A SWRL rule is composed of two parts, an antecedent and a consequent where both the antecedent and the consequent consist of a positive conjunction of atoms. Since SWRL rules are expressed in terms of OWL concepts, atoms can be either individuals, properties or classes defined within the knowledge base. This feature of SWRL is particularly important in SSHS because, in rule definition, IoT sensor data and knowledge can be composed together in a seamless way, thus allowing to achieve a higher level of abstraction. For example, a rule to automatically turn on a lamp when the natural light drops could be written as shown in Listing 4.1 regardless of how the low light condition of the room has been actually determined. For instance, it could be either obtained by analysing the output of a physical photocell (this would be the only possible way with a trigger-action system) or inferred using the information contained in an ontology given the period of the year and the current time. Alternatively, the same information could be achieved by consulting a web service.

```
lowLightLevel(Room) -> switchOn(Lamp)
```

Listing 4.1: A SWRL rule tha can be used to switch on a lamp in case of low light condition inside a room.

This approach overcomes the limitations of the simple trigger-action model; the higher level of abstraction in rule definition helps significantly to tackle the complexity of defining advanced home automation scenarios. Most importantly, automated reasoning capabilities provided by OWL and SWRL dramatically increase the overall degree of automation that can be potentially achieved by the system.

In trigger-action programming, to check whether scenario conditions are met two types of triggers can be employed: event triggers and state triggers [52]. Event triggers refer to when an asynchronous event occurs, that is when a certain condition becomes true at a particular instant in time. For example, when a button is pressed or when a presence sensor detects a person entering a room. State triggers occur when a condition is true over a period of time. For instance, the condition that it is raining outside or the temperature is above a certain threshold.

In the SSHS the only way to determine where scenario conditions are met is to analyse the current status of the house represented by the RDF graph that is constructed during the update phase. Since the home status RDF graph is sampled at regular intervals, state triggers are naturally supported, but asynchronous events can not activate any immediate response. However, in SSHS, event triggers can be easily simulated by checking whether the event condition has occurred in the near instant using the timestamp provided by the IoT devices. For example, the rule in Listing 4.2 can be used to automatically turn on the garden lights when a car passing through the gate is detected by a photocell.

```
hasClosedState(photocell)
^ hasTimeStamp(photocell, ?ts)
^ currentTime(?ct)
^ swrlb:subtract(?ct, ?ts, ?delta)
^ swrlb:greaterThan(?delta, 0)
^ swrlb:lessThan(?delta, 10000)
-> switchOn(gardenLight)
```

Listing 4.2: This SWRL rule switches on the garden lights if the photocell has detected the passage of a car within the last ten seconds.

The above rule checks whether the difference between the current time and the instant reported in the time stamp is below a certain threshold. The rule can be simplified by defining another rule which generalises this concept as shown in Listing 4.3.

```
saref:hasTimeStamp(?device, ?ts)
^ currentTime(?ct)
^ swrlb:subtract(?ct, ?ts, ?delta)
^ swrlb:greaterThan(?delta, 0)
^ swrlb:lessThan(?delta, 10000)
-> tenSecondsEvent(?device)

tenSecondsEvent(gardenLight) -> switchOn(gardenLight)
```

Listing 4.3: The first rule is used to detect that a generic event has happened within the last ten seconds while the second rule makes use of the first one to control the garden lights.

Generally, time stamps enable time reasoning capabilities that are particularly useful for dealing with ordered events. For example, an action is performed only if a person has pressed a button after entering a room and not *vice versa*.

It is worth to note that neither SWRL nor OWL natively provide current time functionality which is essential to simulate event triggers in the SSHS. However, time information can be easily introduced into the system by using virtual IoT devices (i.e., in this case a simulated clock device).

Similarly to triggers, actions can be distinguished into instant actions, sustained actions, and extended actions [52]. Instant actions occur at a particular instant of time such as "turn on the light". Sustained actions are performed as long as a condition holds, for example, "light is on as long as there is someone in the room". Extended actions are performed for a specific time interval, therefore time stamps are needed to accomplish this task. For example, "change the light colour to red for 10 seconds".

After a sustained or extended action it is often necessary to resume the previous state of the device. Since it is not possible to store indefinitely new knowledge within the knowledge base (i.e., the house state graph is rebuilt at each update phase), augmented IoT devices can be employed when the devices do not natively provide a status history or when it is not possible to infer the previous state from the current graph.

OWL and consequently SWRL knowledge bases are monotonic; new knowledge can be added, but existing knowledge can not be retracted or modified. In the SSHS, rules can not directly change the state of the IoT devices. For example, if a lamp has to be turned on, rules classify the respective OWL individual into a desired state which represents the action that should be taken. Only during the update phase actual commands are sent to devices according to the classification results in order to reflect changes. Actual changes to the environment are visible within the knowledge base only after the next update phase has been completed.

4.3 Results

An IoT home automation system is essentially composed of a variable number of devices that include both sensors to perceive the domestic environment and actuators to perform actions on it, connected to a programmable unit that implements the control logic (i.e., the domotic scenarios). The majority of traditional IoT home automation systems adopt the trigger-action model as a programming paradigm for defining scenarios, according to which every time certain conditions are met, a specific action is executed.

The SSHS extends the trigger-action model by introducing automated reasoning capabilities in the process. In this system, domotic scenarios are expressed by means of SWRL rules that combine knowledge with trigger-action definitions. Therefore, the automatic actions are not only determined based on sensor data, but are the result of logical inferences enabled by the underlying semantics provided by the SAREF ontology. The aim of this approach is to achieve greater expressivity and a higher level of abstraction needed to build knowledge-enabled and reasoning-capable home automation systems, so that the potential offered by IoT devices can be fully exploited.

In order to show the feasibility of the proposed approach, on our prototype several different knowledge-involving domotic scenarios that operate under different conditions with different kinds of IoT devices have been tested. These use cases also are meant to demonstrate that scenarios in SSHS can be based on web standards and public ontologies and implement well-defined reasoning without the necessity of *ad hoc* control programs or even *ad hoc* ontologies.

Moreover, the system performance were evaluated by measuring the processing time taken by the reasoner to evaluate rules with different numbers of devices involved in order to demonstrate the efficiency of the SSHS in realistic settings.

Experiments were conducted using a simulated house environment. IoT devices output data were synthetically generated using a Python script. This testing method is particularly convenient since it is economical, significantly speeds up the process of recreating the desired experimental conditions and simplifies the analysis of the resulting system behaviour. Moreover, it is worth to note that since the semantics provided by the SAREF ontology makes data device-independent, synthetic generated IoT output graphs do not lack any relevant information that might be acquired in a real life setting.

An example of a simulated domestic environment that was used as a testbed to carry out experiments is shown in Figure 4.3. In this case, the environment consists of a 6 room flat plus a terrace, equipped with 29 IoT devices including light switches, door/windows switches, thermometers and a presence detection sensor. An excerpt of the corresponding RDF graph model of the house is shown in Listing 4.4. As it can be seen, rooms are modelled as OWL individuals that belong to specific classes according to their functions within the house. SAREF4BLDG[4], an extension of the SAREF core ontology, has been employed for this purpose. For example, the terrace is represented by means of an OWL individual that belongs to the class `s4bldg:BuildingSpace` to indicate that it is an outdoor ambient.

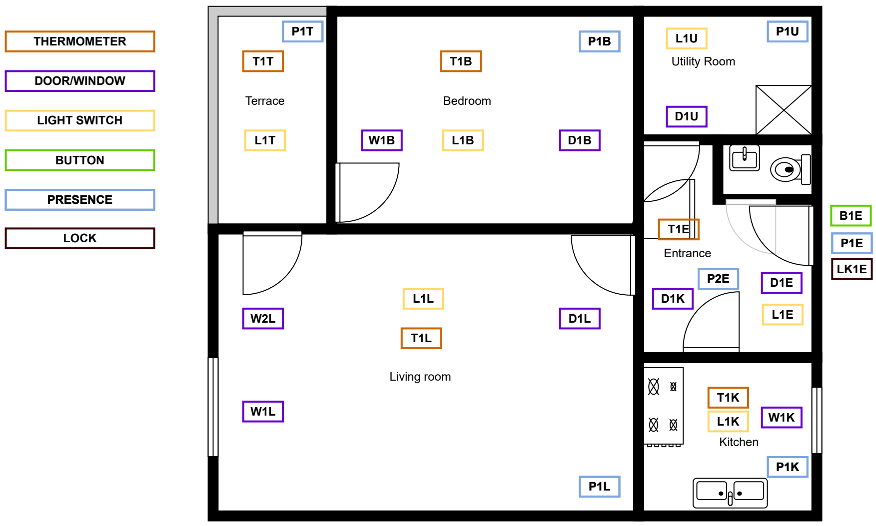

Figure 4.3: The schema represents the floor-plan of typical domestic environment equipped with various domotic IoT devices.

```
ex:Building rdf:type s4bldg:Building .
```

[4]SAREF4BLDG ontology: https://saref.etsi.org/saref4bldg/

```
ex:Building s4bldg:hasSpace ex:House .
ex:House rdf:type s4bldg:BuildingSpace .
ex:House rdf:type saref:FeatureOfInterest .
ex:House s4bldg:hasSpace ex:LivingRoom .
ex:LivingRoom rdf:type s4bldg:BuildingSpace .
ex:LivingRoom rdf:type saref:FeatureOfInterest .
ex:House s4bldg:hasSpace ex:Terrace .
ex:Terrace rdf:type s4bldg:BuildingSpace .
ex:Terrace rdf:type saref:FeatureOfInterest .

ex:LivingRoom s4bldg:contains ex:Door .
ex:Door rdf:type s4bldg:BuildingObject .
ex:LivingRoom s4bldg:contains ex:Window .
ex:Window rdf:type s4bldg:BuildingObject .
s4bldg:BuildingObject rdfs:subClassOf s4bldg:PhysicalObject .

ex:LivingRoom s4bldg:contains ex:LightSwitch .
ex:LightSwitch rdf:type saref:Device .
saref:Device rdfs:subClassOf s4bldg:PhysicalObject .
ex:LightSwitch rdf:type saref:Actuator .

ex:LivingRoom s4bldg:contains ex:Thermometer .
ex:Thermometer rdf:type saref:Device .
ex:Thermometer rdf:type saref:TemperatureSensor .

ex:LivingRoom s4bldg:contains ex:PresenceDetector .
ex:PresenceDetector rdf:type saref:Device .
ex:PresenceDetector rdf:type saref:Sensor .
```

Listing 4.4: An excerpt of the RDF graph that represents the spatial features of the domestic environment shown in Figure 4.3.

Static knowledge about the environment and any other relevant information that cannot be inferred from the IoT data sources, such as the aforementioned spatial features of the house, can be introduced into the SSHS as configuration data in RDF format. There are no constraints on the quantity and the type of information that can vary significantly and strictly depends on the specific scenarios that make use of it.

An example of an RDF graph representing an IoT data source and its associated measurement is shown in Listing 4.5. The graph regards a thermometer that is physically located on the terrace. The measurement value is reported along with the unit and the timestamp of the gauging instant in ISO-8601 format and Unix time.

```
ex:T1T rdf:type saref:TemperatureSensor ;
       saref:makesMeasurement ex:T1T_m1 ;
       s4bldg:isContainedIn ex:Terrace .

ex:T1T_m1 rdf:type saref:Measurement ;
         saref:hasValue "25.5"^^xsd:float ;
         saref:isMeasuredIn om:degree_Celsius ;
         saref:hasTimestamp"2020-12-02T14:30:00"^^xsd:dateTime ;
         smart:hasUnixTimestamp "1606919400"^^xsd:integer .
```

Listing 4.5: An excerpt of RDF graph that represents the output of an IoT thermometer annotated according to the SAREF ontology. The graph contains information about the physical collocation of the device within the house as well as the measurement in degree Celsius and the timestamp.

During normal operation, the SSHS cyclically aggregates the various IoT data sources to construct an RDF graph that represents the current status of the domestic environment. Rules are then executed over this graph and the inferred knowledge is added back to it. Lastly, the system inspects the resulting graph to send, if necessary, commands to the actuator devices. For these tests, the last phase was omitted and the resulting graph was directly analysed using the Protege tool.

4.3.1 Energy Conservation Monitoring Scenario

Often, homeowners are unaware of the costs of some domestic energy wasting behaviours they involuntarily adopt, such as forgetting the windows open or leaving them open for too long a period of time. Therefore, monitoring in-home energy wasting in order to provide tenants immediate feedback can be a crucial feature for actuating an efficient energy wasting reduction in an IoT home automation system [216].

A domotic scenario that assesses the energy wasting level of a room according to the state of the windows (i.e., either open or closed) was implemented. Top-down proceeding, the monitoring system consists of two main rules, shown in Listing 4.6, that classify the energy wasting level into two states: GreenEnergyState if there is no energy wasting (i.e., windows in the room are closed) or RedEnergyState if there is energy wasting (i.e., windows in the room are open).

```
smarthouse:ClosedWindow(?window)
^ s4bldg:isContainedIn(?window, ?room)
-> smarthouse:GreenEnergyState(?room)

smarthouse:OpenWindow(?window)
^ s4bldg:isContainedIn(?window, ?room)
-> smarthouse:RedEnergyState(?room)
```

Listing 4.6: SWRL rules for classifying the energy wasting level in a room.

For each window of the house, first its state is detected. Then, the location of the window within the house is retrieved, and the corresponding room is finally classified accordingly.

A possible way to detect whether a window is open or closed could be achieved by installing a door switch IoT sensor on the window frame. For example, the *ClosedWindow* and the *OpenWindow* classification can be determined by retrieving the device state using the rules in Listing 4.7.

```
smart:Window(?window)
^ saref:hasState(?window, ?state)
^ saref:CloseState(?state)
-> smarthouse:ClosedWindow(?window)

smart:Window(?window)
^ saref:hasState(?window, ?state)
^ saref:OpenState(?state)
-> smarthouse:OpenWindow(?window)
```

Listing 4.7: The SWRL rules are used to detect whether a window is open using a switch sensor state .

Since it cannot be inferred that a sensor is either installed on a window or a door, it is necessary to explicitly specify it as configuration data. This can be done either through the device settings or using rules. For example, the rule in Listing 4.8 states that the OWL individual *W1L* is a door switch installed on a window.

```
saref:DoorSwitch(W1L) -> Window(W1L)
```

Listing 4.8: The SWRL rule asserts that a DoorSwitch sensor *W1L* is installed on a window.

The location of the window can be inferred through the location of the sensor (if it has been indicated in the device settings).

However, not always an open window causes heat loss, for example when the difference between the indoor and outdoor temperature is negligible. Therefore, to achieve more precision, rules can be modify so that the temperature is taken into account as in Listing 4.9.

```
smarthouse:OpenWindow(?window)
^ s4bldg:isContainedIn(?window, ?room)
```

```
^ smarthouse:hasIntExtTempDifference(?room, delta)
^ swrlb:greaterThan(?delta, 0.5)
-> smarthouse:RedEnergyState(?room)
```

Listing 4.9: SWRL rules for classifying the energy wasting level in a room by taking into account several factors.

The internal temperature can be acquired by installing a thermometer inside the room, while the external temperature can be acquired by either installing a thermometer outside (e.g., on the terrace) or using an IoT virtual device that wraps an online weather web service. When the two measurements are available the difference can be easily obtained as shown in Listing 4.10:

```
smarthouse:hasIndoorTemperature(?room, ?internalTemp)
^ smarthouse:hasOutdoorTemperature(?house, ?externalTemp)
^ swrlb:subtract(?internalTemp, ?externalTemp, ?delta)
-> smarthouse:hasIntExtTempDifference(?room, delta)
```

Listing 4.10: SWRL rule for calculating the difference between the internal temperature and the external temperature.

Eventually, once an energy wasting situation has been detected, remedial actions could be performed such as warning the tenants or automatically turning off the heating system in the room.

4.3.2 Visual Cueing System Scenario

Largely widespread in the modern domotic landscape, IoT smart light devices are inexpensive colour-changing LED light bulbs remotely controllable. Initially intended for ambient lighting enhancement, smart bulbs can be efficiently used as means to provide visual cues in IoT home automation systems. For example, if tenants are watching the TV or speaking on the phone, to signal the end of the washing machine cycle, the room light could be slightly turned to a blue-like colour instead of relying on an annoying buzzer sound. Most importantly, such a pervasive visual cueing system can potentially result in an efficient low-cost assistive technology in case of hearing impaired tenants [90].

A domotic scenario that exploits the colour-changing features of the IoT house lighting system to signal the presence of a visitor at the entrance door was designed.

```
smarthouse:VisitorAtTheDoor(Entrance)
^ smarthouse:Tenant(?tenant)
^ smarthouse:isLocatedIn(?tenant, ?room)
^ saref:LightSwitch(?light)
^ smarthouse:isLocatedIn(?light, ?room)
-> smarthouse:TemporaryRedLightColourState(?light)
```

Listing 4.11: SWRL rule that signals the presence of a visitor by turning the colour of indoor light to red.

The rule in Listing 4.11 turns the lights colour to red when someone has pressed the doorbell or is standing close to the entrance door. Tenant localisation within the house is performed in order to avoid unnecessary light switching (i.e., only lights in occupied rooms are involved). Alternatively, if indoor tenant localisation is not possible (e.g., due to the lack of presence sensors in every room or it cannot be inferred in another way), a restricted set of light bulbs can be specifically designated for this purpose.

To detect a visitor at the entrance door, either a presence/motion sensor mounted on the door frame or a smart button can be used. When a presence sensor is used, the rule shown in Listing 4.12 classifies the entity `Entrance` into the state `VisitorAtTheDoor` according to sensor state.

```
saref:PresenceSensor(P1E)
^ saref:hasState(P1E, OnState)
-> smarthouse:VisitorAtTheDoor(Entrance)
```

Listing 4.12: This SWRL rule is used to detect the presence of a visitor at the door using a volumetric sensor.

Similarly, the same operation has to be performed when a doorbell button (i.e., a smart button) is employed. However, since a button is kept pressed only for a few instants of time, a timing mechanism is needed since the entity `Entrance` should persist in the `VisitorAtTheDoor` state for a certain number of seconds after the button has been released. This can be achieved by checking the button timestamp, that is, the visitor presence is detected until 10 seconds have passed since the button press occurred. To retrieve the current time, a virtual IoT device `MainClock` is used. An example of such a rule is provided in Listing 4.13

```
smarthouse:hasUnixTimestamp(MainClock, ?clockTime)
^ smarthouse:SmartButton(B1E)
^ smarthouse:hasUnixTimestamp(B1E, ?doorbellTime)
^ swrlb:subtract(?elapsedTime, ?clockTime, ?doorbellTime)
^ swrlb:lessThan(?elapsedTime, 10000)
-> smarthouse:VisitorAtTheDoor(Entrance)
```

Listing 4.13: SWRL rule to detect the pressure of the doorbell button using the device timestamp.

The proposed scenario can be easily extended or modified. For example, tenants could receive an alert on their mobile phones if they are not inside and the visual signalling can be automatically disabled during sleep time.

4.3.3 Weather Based Domotic Scenario

IoT weather stations are devices that provide information about the local external environment, including temperature, humidity, wind speed, and rain conditions. In home automation, local climate data can be exploited in several ways, for example, to automatically regulate the curtain position according to natural light and retract it in case of strong wind. One disadvantage of IoT weather stations is that they are expensive equipment, especially compared to other common IoT domotic devices. However, in the SSHS, a physical IoT weather station can be easily replaced by a virtual IoT device that retrieves the same information from an online weather web service.

A domotic scenario that, based on current weather conditions, automatically disables the garden irrigation system in case of rain, thus reducing unnecessary water consumption was designed.

For this scenario, a virtual IoT device that acts as a weather station has been implemented. All the information that the virtual device provides is retrieved from the OpenWeather web service [189]. OpenWeather offers access to current weather data for any location specified by geographic coordinates or city name. Data include information about the weather conditions (e.g., rain, snow, etc.), temperature, humidity, wind speed, and in particular the rain volume for the last 3 hours.

Basically, the virtual device consists in a Python script that queries the OpenWeather web service and translates the JSON result into RDF. An excerpt of the output graph is shown in Listing 4.14.

```
ex:WS1 a smart:SmartWeatherStation ;
      saref:hasTimestamp "2020-12-02T14:30:00"^^xsd:dateTime;
      smart:hasUnixTimestamp "1606919400"^^xsd:integer ;
      smart:hasSunriseTime 1634018420 ;
      smart:hasSunsetTime 1634057589 ;
      smart:hasTemperature 9.29 ;
      smart:hasWeatherCondition smart:WeatherConditionRain;
      smart:hasRain3h 3.
```

Listing 4.14: An excerpt of the output RDF graph of the weather base station implemented through a virtual IoT device.

It is supposed that the irrigation system is activated every day at a fixed time. The rules in Listing 4.15 prevent the electromechanical valve from opening if the rainfall amount in the last 3 hours is greater than 2mm.

```
smart:SmartWeatherStation(WS1)
^ smart:hasRain3h:(WS1, ?rainfall)
^ swrlb:greaterThan(?rainfall, 2)
-> smart:keepClosedState(VALVE1)
```

Listing 4.15: This SWRL rule prevent the irrigation system to water the garden if the amount of rain in the last 3 hour exceeds 2mm.

Similarly, the irrigation system should be disabled if it has started raining during its functioning. This can be obtained by adding another rule as shown in Listing 4.16.

```
smart:SmartWeatherStation(WS1)
^ smart:hasWeatherCondition(WS1, WeatherConditionRain)
-> smart:keepClosedState(VALVE1)
```

Listing 4.16: This SWRL rule prevents the irrigation system to water the garden while it is raining.

It has to be noted that two separate rules are necessary since SWRL does not support atom disjunction.

4.3.4 Performance Evaluation

The main advantage of the SSHS over traditional systems based on the trigger-action model is the capability of performing automated reasoning over sensor data and formalised knowledge provided by SWRL rules and OWL ontologies. Knowledge-based systems enabled by DL-OWL reasoners necessitate higher computational capacity compared to systems implemented using general purpose languages such as C++ or Java. Nevertheless, home automation systems are real-time applications that require logic instructions to be executed in a minimal amount of time so as not to cause unwanted delays in automatic actions. Therefore, it is essential that the SSHS completes the inferential process in appropriate time to make it suitable to operate in a real-setting environment.

In order to asses the time performance of the SSHS, similarly to what has been done by Zhai et al. in [262], a set of rules were executed varying the number of devices involved in each experiment repetition, and the average processing time (i.e., the time that the reasoner takes to draw all the inferences) for each configuration was measured. The employed rule-set comprehends: 2 rules that determine whether a room is occupied or unoccupied according to a sensor state (i.e., classification task); 5 rules that classify the air quality into five levels based on the concentration of the CO2 in the room [5] (i.e., a task that require significant numerical comparisons); 2 rules that make use of the preceding results and an OWL ontology to automatically open the windows in the room in case of high concentration of CO2 if people are present (i.e., a task than involve a chain of inferences and additional OWL axioms). An excerpt of the rule-set is provided in Listing 4.17.

```
smart:CO2Meter(?device)
^ smart:isLocatedIn(?device, ?room)
^ saref:makesMeasurement(?device, ?measurement)
^ saref:hasValue(?measurement, ?value)
^ swrlb:greaterThan(?value, 400)
^ swrlb:lessThanOrEqual(?value, 1000)
-> smart:CO2GreenLevel(?room)

smart:PresenceSensor(?device)
^ saref:hasState(?device, example:OnState)
```

[5]Wisconsin Department of Health Services. Health effects produced by exposure to CO2. https://www.dhs.wisconsin.gov/chemical/carbondioxide.htm

```
^ smart:isLocatedIn(?device, ?room)
-> smart:OccupiedRoom(?room)

smart:CO2UnsafeLevel(?room)
^ smart:OccupiedRoom(?room)
-> smart:WindowsOpen(?room)
```

Listing 4.17: An excerpt of the set of rules employed to measure the average processing time.

The status graph, over which the inferences are drawn, contains the measurements collected by a presence detection sensor and a CO2 meter. Both the sensors are installed in the same room. The presence sensor indicate whether the room is occupied through an `OffState-Onstate` indication, while the CO2 meter measures the CO2 concentration in the room in ppm. Measurement values and presence states were randomly generated using a Python script. The number of the rooms was increased to vary the number of devices involved in each run of the experiment. Listing 4.18 provides an example of the status graph generated to test the system time performance.

```
ex:CO2_1 rdf:type smart:CO2 ;
     saref:makesMeasurement ex:CO2_1_m1 ;
     smart:isLocatedIn ex:R1 .

ex:CO2_1_m1 rdf:type saref:Measurement;
                    saref:hasValue "2186"^^xsd:float ;
     saref:isMeasuredIn om:partsPerMillion ;
     saref:hasTimestamp "2020-12-02T14:30:00"^^xsd:dateTime .

ex:PRESENCE_1 rdf:type smart:SmartPresence ;
     saref:hasState ex:OnState ;
     smart:isLocatedIn ex:R1 .
```

Listing 4.18: An excerpt of the the status graph generated to test the system time performance.

For each repetition of the experiment the number of devices involved was increased. The inference process has been executed three times for each configuration and the average time calculated. The tests were performed using Protege 5.5.0 on a MacBook Pro equipped with an Intel Core i7 (2,2GHz, 6-Core) processor and 16GB (2400 MHz DDR4) of RAM. It is worth to note that existing semantic reasoners are too resource-intensive to be directly ported on resource-constrained devices (such as Raspberry or Arduino) without further engineering efforts. However, promising solutions are currently under investigation [37].

Figure 4.4 shows the results of the experiments. The execution time is specified in milliseconds. As it can be seen from the graph, for a number of devices between 2 and 200 the execution time does not exceed 300ms which can be considered optimum for real-time operation in a domotic environment where the number of devices of a typical deployment is expected to be in the range of 50 to 100[252]. This result is significant when compared to the high latency and variability of the popular IFTTT trigger-action system that has been experimentally estimated in the order of seconds [172].

4.4 Discussions

The ubiquity and pervasiveness of the IoT devices in ordinary households offers important opportunities for the home automation sector in many crucial areas such as energy preservation, home safety and living assistance. Notwithstanding, the potential of these devices is still largely underexploited due to the poor data interoperability and the limited expressivity of the most commonly adopted paradigms for programming domotic scenarios.

The aim of this study was to show that SW technologies can constitute a viable solution not only for tackling the problem of data and device heterogeneity, but also for defining more complex home automation scenarios that can better exploit the potential of IoT technology. To this purpose, it was

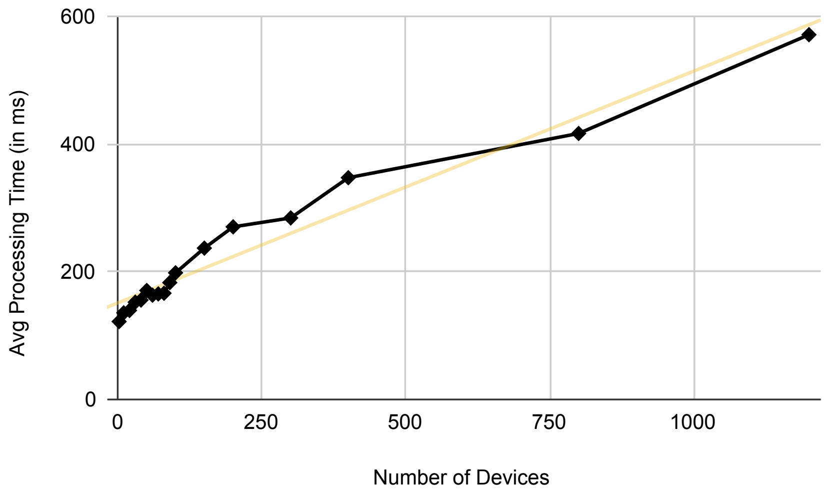

Figure 4.4: The graph displays the average processing time (in milliseconds) for a number of devices that ranges from 2 to 1200.

developed a knowledge-based IoT home automation system that can aggregate and combine IoT sensor data with external heterogeneous data sources and analyse them according to formal knowledge expressed by means of OWL ontologies and SWRL rules. In particular, it has been demonstrated how in such a system the SWRL language can extend and overcome the limitations of the popular trigger-action model by introducing inferential reasoning capabilities directly in domotic scenario definition. The variety of the experiments that were carried out using SAREF as the main reference ontology proved the feasibility of the proposed approach and applicability of it to a standard and well validated context.

The proposed system opens up to a plethora of possible domotic scenarios that can be implemented ranging from simple actions to more complex automation tasks that make use of common sense knowledge.

Other benefits of the proposed approach include the high degree of explainability of the processes involved. This feature assumes particular importance when home automation is employed to realise in-home healthcare facilities, for example, for elderly care or assisted living. Moreover, due to the high degree of customisation and the overall flexibility, the system could be easily re-adapted for use in other IoT domains where complex automation logic is needed.

However, it is worth to note that at present, even though reuse is partially possible, scenarios require rules to be manually written. Manual rule-writing is not a trivial task but a time consuming and error prone process especially when scenario complexity significantly increases. Therefore, alternative ways to facilitate and potentially automatise the rule creation should be explored. Future research will focus on how to apply machine learning techniques to ease the process of creating rule-based home automation scenarios. Other interesting research directions should address the problems of diagnosis and fault tolerance.

Chapter 5

Conclusions

The high heterogeneity of devices and data sources that characterises the IoT landscape provides both challenges and opportunities to develop innovative systems and services. However, the ubiquitous presence of data silos and the poor semantic interoperability among IoT devices constitute a significant obstacle to achieve this goal. Furthermore, in order to fully exploit the potential of modern IoT devices (i.e., achieving actionable knowledge from the collected information), IoT data have to be analysed using appropriate artificial intelligence techniques such as automated reasoning. Indeed, current IoT systems still highly rely on human intervention for interpreting the collected information in order to take actions, whereas a higher level of autonomy could be reached by analysing sensors data according to some formalised knowledge and, potentially, in conjunction with disjoint-domain information sources.

In this thesis work SW technologies approaches have been investigated to address both the data integration and reasoning aspects in modern IoT systems. The IFO ontology was developed in order to overcome the issues of data silos and support semantic interoperability in the IoT fitness and wellness domain. Essentially, the IFO ontology provides a formal representation of the most important and common concepts in the domain, and the relationships among them. The use of the OWL language allows the ontology to be efficiently employed and easily integrated in a huge number of different applications. In particular the applications that are required to deal with information coming from multiple heterogeneous devices and need to analyse data in conjunction with other external information sources. In this thesis study, the IFO ontology was also used as the core component for the development of the LOD portal described in Chapter 3. Sharing health datasets in an efficient way is a crucial service to the scientific community. Previous platforms and data-sharing portals were mere repository containers, data from IoT devices were only collected and store. The LOD portal presented in this thesis homogenises the heterogeneous health datasets that users upload and share them according the Linked Data recommendations.

Besides easing the integration of disjoint information sources, another crucial feature of SW technologies is to enable automated reasoning capabilities over the semantically annotated data. In a context dominated by multiple-facets of heterogeneity such as the IoT field, automated reasoning constitute an important means to tackle the overall complexity of data analysis.

Modern smart houses consist of an interconnected set of devices that sense the home environment and perform actions upon it. The majority of the commercial IoT home automation systems are based on the simple trigger-action model for defining the control logic. Even though trigger-action systems through if-then rules provide a simple and efficient programming method, they lack the expressivity and a higher abstraction level needed to define more complex scenarios. Previous studies that have investigated the use of SW technologies in IoT home automation focused solely on specific domotic tasks (e.g., energy management) and proposed solutions that strictly depend on specific *ad hoc* written ontologies. In this thesis work, a novel methodology that employs the SWRL language as an ontology-agnostic programming paradigm for IoT home automation systems was investigated. The proposed approach extends the trigger-action model with the possibility of embedding knowledge directly and seamlessly into rules, thus overcoming the limitations of the trigger-action systems. Additionally, the SSHS, described in Chapter 4 was developed to support the execution of the semantic rules defined scenarios. The feasibility and the efficiency of the proposed approach in realistic real-life settings have been demonstrated through numerous experiments.

The central idea of this thesis is that the potential offered by IoT devices can only be fully exploited if the heterogeneous collected data are integrated with information coming from disjoint-domains sources and the cognitive analysis process is efficiently automatised. The aim of this thesis work was to investigate a SW-based approach to address both the issues. Data integration and reasoning are two complementary aspects since reasoning to be effective need to operate on multiple information sources and data integration make this possible by making them available in a homogenised way. The healthcare and the domotic IoT devices have been chosen since they represent an emblematic example of the multiple-facets issues of heterogeneity in the IoT technology. Indeed, these are two IoT ecosystems in which the variety of the devices and the multitude of different kinds of data collected constitute their richness and at the same time the main challenges. To conclude, it is worth to note that the applicability of the findings and the methodologies presented in this thesis are not limited to these two fields only, but they can be easily and efficiently extended to other IoT domains as well.

5.0.1 Future Research Directions

In this thesis work, symbolic approaches have been adopted as the main means to process IoT data in order to achieve actionable knowledge. Indeed, automated reasoning (which basically consists of drawing logical inferences from available information) is natively supported by SW technologies and provides crucial features such as the ability to explain the reached conclusions. Additionally, symbolic systems provide a human understandable computation flow which makes them intuitive to design and offer a high degree of modularity and interoperability. For example, SW rules are self-contained knowledge units that can be easily transferred from a knowledge base to another. Moreover, symbolic systems are a convenient approach to model general abstract problems which is particularly useful to tackle the complexity that characterises IoT technology. However, symbolic methods are not well-suitable for cases where data are missing or noisy as it often happens in IoT systems. Further, hand-coded rules and manual knowledge modelling require lot of human involvement since they are complex and time consuming tasks.

Contrary to symbolic systems, sub-symbolic systems (e.g., artificial neural networks) are based on statistical methods and are better suitable for tasks that require predictions, clustering, pattern classification and recognition of entities. Essentially, sub-symbolic techniques establish correlations between input and output variables with minimal or no human intervention. Therefore, they require less knowledge upfront and are more robust against incomplete and noisy data. However, one significant disadvantage of sub-symbolic systems is the lack of explainability which is a crucial requirement in healthcare. Moreover, statistical based models require large amounts of high quality training data not to result in biased outcomes.

Given the aforementioned considerations, it is therefore evident that future research should be oriented towards the integration (or the combination) of symbolic systems with sub-symbolic approaches [60]. Studies in the literature suggest that hybrid approaches are suitable for dealing with large amounts of heterogeneous data [106]. The importance of intertwining perceptual computing with semantic computing and cognitive computing in IoT systems have also been highlighted in [226, 227].

The SSHS presented in this thesis could be extended through perceptual computing in order to exploit contextual information and provide personalised rules adjustments. Data mining techniques could be employed to detect usage patterns and automatically preempt tenants actions. Machine learning algorithms could be leveraged to facilitate and automatise the rule design process or for rule extraction. Probabilistic logic programming paradigms could enrich rules with probabilities so that scenarios can be defined also under conditions of uncertainty.

Bibliography

[1] Apache jena fuseki.
https://jena.apache.org/documentation/fuseki2/
(Last Accessed: 2022-08-16).

[2] Ifttt.
https://ifttt.com
(Last Accessed: 2022-08-16).

[3] Javaserver pages technology.
http://www.oracle.com/technetwork/java/index-jsp-138231.html
(Last Accessed: 2022-08-16).

[4] Kaggle.
https://www.kaggle.com/
(Last Accessed: 2022-08-31).

[5] The open definition.
http://opendefinition.org
(Last Accessed: 2022-08-31).

[6] Open humans project.
https://www.openhumans.org/
(Last Accessed: 2022-08-31).

[7] Physiobank.
https://www.physionet.org/physiobank
(Last Accessed: 2022-08-31).

[8] R2rml: Rdb to rdf mapping language.
https://www.w3.org/TR/r2rml/
(Last Accessed: 2022-08-31).

[9] Researchkit.
https://developer.apple.com/researchkit/
(Last Accessed: 2022-08-31).

[10] Swrl built-in specification.
http://www.daml.org/rules/proposal/builtins.html
(Last Accessed: 2022-08-31).

[11] Cisco visual networking index: Global mobile data traffic forecast update, 2016â2021 white paper, 2017.

[12] T. Abbas, V.-J. Khan, and P. Markopoulos. Investigating the crowdâs creativity for creating on-demand iot scenarios. *International Journal of Human–Computer Interaction*, 36(11):1022–1049, 2020.

[13] D. Abián, F. Guerra, J. Martínez-Romanos, and R. Trillo-Lado. Wikidata and dbpedia: a comparative study. In *International KEYSTONE Conference on Semantic Keyword-Based Search on Structured Data Sources*, pages 142–154. Springer, 2017.

[14] K. Abouelmehdi, A. Beni-Hessane, and H. Khaloufi. Big healthcare data: preserving security and privacy. *Journal of Big Data*, 5(1):1, 2018.

[15] B. Adida, M. Birbeck, S. McCarron, and S. Pemberton. Rdfa in xhtml: Syntax and processing. *Recommendation, W3C*, 7, 2008.

[16] D. B. Adriano, W. A. C. Budi, et al. Iot-based integrated home security and monitoring system. In *Journal of Physics: Conference Series*, volume 1140, page 012006. IOP Publishing, 2018.

[17] I. D. Agranat. Engineering web technologies for embedded applications. *IEEE Internet Computing*, 2(3):40–45, 1998.

[18] E. Agu, P. Pedersen, D. Strong, B. Tulu, Q. He, L. Wang, and Y. Li. The smartphone as a medical device: Assessing enablers, benefits and challenges. In *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2013 10th Annual IEEE Communications Society Conference on*, pages 76–80. IEEE, 2013.

[19] V. Ahanathapillai, J. D. Amor, Z. Goodwin, and C. J. James. Preliminary study on activity monitoring using an android smart-watch. *Healthcare technology letters*, 2(1):34–39, 2015.

[20] M. Alaa, A. A. Zaidan, B. B. Zaidan, M. Talal, and M. L. M. Kiah. A review of smart home applications based on internet of things. *Journal of Network and Computer Applications*, 97:48–65, 2017.

[21] H. Alemdar and C. Ersoy. Wireless sensor networks for healthcare: A survey. *Computer Networks*, 54(15):2688–2710, 2010.

[22] H. T. Alvestrand. Tags for the identification of languages. 2001.

[23] F. Amardeilh. Semantic annotation and ontology population. *Semantic Web Engineering in the Knowledge Society*, page 424, 2008.

[24] D. Andročec, M. Novak, and D. Oreški. Using semantic web for internet of things interoperability: A systematic review. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 14(4):147–171, 2018.

[25] A. A. Atienza and K. Patrick. Mobile health. *American journal of preventive medicine*, 40(5):S151–S153, 2011.

[26] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[27] F. Baader. *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003.

[28] S. Balakrishnan, H. Vasudavan, and R. K. Murugesan. Smart home technologies: A preliminary review. In *Proceedings of the 6th International Conference on Information Technology: IoT and Smart City*, pages 120–127, 2018.

[29] M. Bang, K. Solnevik, and H. Eriksson. The nurse watch: design and evaluation of a smart watch application with vital sign monitoring and checklist reminders. In *AMIA Annual Symposium Proceedings*, volume 2015, page 314. American Medical Informatics Association, 2015.

[30] P. Barnaghi, P. Cousin, P. Maló, M. Serrano, and C. Viho. Simpler iot word (s) of tomorrow, more interoperability challenges to cope today. *River publishers series in communications*, page 277, 2013.

[31] P. Barnaghi, W. Wang, C. Henson, and K. Taylor. Semantics for the internet of things: early progress and back to the future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1):1–21, 2012.

[32] B. R. Barricelli and S. Valtolina. Designing for end-user development in the internet of things. In *International symposium on end user development*, pages 9–24. Springer, 2015.

[33] D. Beckett, T. Berners-Lee, and E. Prudâhommeaux. Turtle-terse rdf triple language. *W3C Team Submission*, 14(7), 2008.

[34] D. Beckett and B. McBride. Rdf/xml syntax specification (revised). *W3C recommendation*, 10(2.3), 2004.

[35] F. Belleau, M.-A. Nolin, N. Tourigny, P. Rigault, and J. Morissette. Bio2rdf: towards a mashup to build bioinformatics knowledge systems. *Journal of biomedical informatics*, 41(5):706–716, 2008.

[36] J. Bennett, O. Rokas, and L. Chen. Healthcare in the smart home: A study of past, present and future. *Sustainability*, 9(5):840, 2017.

[37] A. Bento, L. Médini, K. Singh, and F. Laforest. Do arduinos dream of efficient reasoners? In *European Semantic Web Conference*, 2022.

[38] A. Berglund, S. Boag, D. Chamberlin, M. F. Fernández, M. Kay, J. Robie, and J. Siméon. Xml path language (xpath). *World Wide Web Consortium (W3C)*, 2003.

[39] M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor. Iot-lite: a lightweight semantic model for the internet of things. In *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pages 90–97. IEEE, 2016.

[40] T. Berners-Lee. Design issues: Linked data (2006). *URL http://www.w3.org/DesignIssues/LinkedData.html*, 2011.

[41] T. Berners-Lee, R. Cailliau, A. Luotonen, H. F. Nielsen, and A. Secret. The world-wide web. *Communications of the ACM*, 37(8):76–82, 1994.

[42] T. Berners-Lee et al. Semantic web road map, 1998.

[43] T. Berners-Lee, J. Hendler, O. Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.

[44] M. J. Bietz, C. S. Bloss, S. Calvert, J. G. Godino, J. Gregory, M. P. Claffey, J. Sheehan, and K. Patrick. Opportunities and challenges in the use of personal health data for health research. *Journal of the American Medical Informatics Association*, 23(e1):e42–e48, 2015.

[45] M. Birbeck and S. McCarron. Curie syntax 1.0–a syntax for expressing compact uris. w3c recommendation, 2009.

[46] P. V. Biron, A. Malhotra, et al. Xml schema part 2: Datatypes.

[47] C. Bizer, T. Heath, and T. Berners-Lee. Linked data-the story so far. *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227, 2009.

[48] C. Bizer, T. Heath, and T. Berners-Lee. Linked data: The story so far. In *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227. IGI Global, 2011.

[49] D. Bonino and F. Corno. Dogont-ontology modeling for intelligent domotic environments. In *International Semantic Web Conference*, pages 790–803. Springer, 2008.

[50] D. Bonino and F. Corno. Rule-based intelligence for domotic environments. *Automation in Construction*, 19(2):183–196, 2010.

[51] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM, 2012.

[52] W. Brackenbury, A. Deora, J. Ritchey, J. Vallee, W. He, G. Wang, M. L. Littman, and B. Ur. How users interpret bugs in trigger-action programming. In *Proceedings of the 2019 CHI conference on human factors in computing systems*, pages 1–12, 2019.

[53] A. Brajdic and R. Harle. Walk detection and step counting on unconstrained smartphones. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 225–234. ACM, 2013.

[54] J. Brank, M. Grobelnik, and D. Mladenić. A survey of ontology evaluation techniques. 2005.

[55] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. Extensible markup language (xml). *World Wide Web Journal*, 2(4):27–66, 1997.

[56] K. Breitman, M. A. Casanova, and W. Truszkowski. *Semantic web: concepts, technologies and applications.* Springer Science & Business Media, 2007.

[57] D. Brickley and R. V. Guha. Rdf vocabulary description language 1.0: Rdf schema. 2004.

[58] D. Brickley and L. Miller. Foaf vocabulary specification 0.99, namespace document 14 january 2014-paddington edition. 2014. http://xmlns.com/foaf/spec.

[59] J. M. Brunetti, S. Auer, R. García, J. Klímek, and M. Nečaskỳ. Formal linked data visualization model. In *Proceedings of International Conference on Information Integration and Web-based Applications & Services*, page 309. ACM, 2013.

[60] R. Calegari, G. Ciatto, and A. Omicini. On the integration of symbolic and sub-symbolic techniques for xai: A survey. *Intelligenza Artificiale*, 14(1):7–32, 2020.

[61] G. Carothers and A. Seaborne. Rdf 1.1 n-triples. w3c recommendation. *World Wide Web Consortium, February*, 2014.

[62] D. Cerri and A. Fuggetta. Open standards, open formats, and open source. *Journal of systems and software*, 80(11):1930–1937, 2007.

[63] C. Y. Chen, J. H. Fu, T. Sung, P.-F. Wang, E. Jou, and M.-W. Feng. Complex event processing for the internet of things and its applications. In *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1144–1149. IEEE, 2014.

[64] G. Chen, T. Jiang, M. Wang, X. Tang, and W. Ji. Modeling and reasoning of iot architecture in semantic ontology dimension. *Computer Communications*, 153:580–594, 2020.

[65] N.-C. Chen, K.-C. Wang, and H.-H. Chu. Listen-to-nose: a low-cost system to record nasal symptoms in daily life. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 590–591. ACM, 2012.

[66] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang. A vision of iot: Applications, challenges, and opportunities with china perspective. *IEEE Internet of Things journal*, 1(4):349–359, 2014.

[67] Y.-H. Chen, M.-J. Tsai, L.-C. Fu, C.-H. Chen, C.-L. Wu, and Y.-C. Zeng. Monitoring elder's living activity using ambient and body sensor network in smart home. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pages 2962–2967. IEEE, 2015.

[68] J. Clark. Xsl transformations (xslt). w3c recommendation, nov. 1999, 1998.

[69] R. J. Cole, D. F. Kripke, W. Gruen, D. J. Mullaney, and J. C. Gillin. Automatic sleep/wake identification from wrist activity. *Sleep*, 15(5):461–469, 1992.

[70] G. O. Consortium et al. The gene ontology (go) database and informatics resource. *Nucleic acids research*, 32(suppl 1):D258–D261, 2004.

[71] U. Consortium et al. *The Unicode Standard, Version 2.0.* Addison-Wesley Longman Publishing Co., Inc., 1997.

[72] W. Consortium et al. Xquery 1.0: An xml query language. *Version*, 1:W3C, 2007.

[73] O. Corcho, M. Fernández-López, and A. Gómez-Pérez. Methodologies, tools and languages for building ontologies. where is their meeting point? *Data & knowledge engineering*, 46(1):41–64, 2003.

[74] F. Corno, L. De Russis, and A. M. Roffarello. A semantic web approach to simplifying trigger-action programming in the iot. *Computer*, 50(11):18–24, 2017.

[75] F. Corno, L. De Russis, and A. M. Roffarello. A high-level semantic approach to end-user development in the internet of things. *International Journal of Human-Computer Studies*, 125:41–54, 2019.

[76] D. Crockford. The application/json media type for javascript object notation (json). 2006.

[77] B. Cuenca-Grau. Owl 1.1 web ontology language tractable fragments, 2007.

[78] L. Da Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on industrial informatics*, 10(4):2233–2243, 2014.

[79] M. C. Daconta, L. J. Obrst, and K. T. Smith. *The Semantic Web: a guide to the future of XML, Web services, and knowledge management.* John Wiley & Sons, 2003.

[80] B. Dan and G. R.V. Rdf schema. - w3c recommendation. https://www.w3.org/TR/rdf-schema/, 2014. Last Accessed: 2018-08-16.

[81] L. Daniele, F. d. Hartog, and J. Roes. Created in close interaction with the industry: the smart appliances reference (saref) ontology. In *International Workshop Formal Ontologies Meet Industries*, pages 100–112. Springer, 2015.

[82] S. Das, S. Sundara, and R. Cyganiak. R2rml: Rdb to rdf mapping language. w3c recommendation 27 september 2012. *Cambridge, MA: World Wide Web Consortium (W3C) (www.w3.org/TR/r2rml)*, 2012.

[83] A. De Nicola, M. Missikoff, and R. Navigli. A proposal for a unified process for ontology building: Upon. In *International Conference on Database and Expert Systems Applications*, pages 655–664. Springer, 2005.

[84] A. De Nicola and M. L. Villani. Smart city ontologies and their applications: A systematic literature review. *Sustainability*, 13(10), 2021.

[85] A. K. Dey, K. Wac, D. Ferreira, K. Tassini, J.-H. Hong, and J. Ramos. Getting closer: an empirical investigation of the proximity of user to their smart phones. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 163–172. ACM, 2011.

[86] V. Dimitrieski, G. Petrović, A. Kovačević, I. Luković, and H. Fujita. A survey on ontologies and ontology alignment approaches in healthcare. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 373–385. Springer, 2016.

[87] A. Dimou, M. V. Sande, P. Colpaert, E. Mannens, and R. Van de Walle. Extending r2rml to a source-independent mapping language for rdf. In *Proceedings of the 2013th International Conference on Posters & Demonstrations Track-Volume 1035*, pages 237–240. CEUR-WS. org, 2013.

[88] A. Dimou, M. Vander Sande, P. Colpaert, R. Verborgh, E. Mannens, and R. Van de Walle. Rml: A generic language for integrated rdf mappings of heterogeneous data. In *LDOW*, 2014.

[89] A. Dimou, M. Vander Sande, J. Slepicka, P. Szekely, E. Mannens, C. Knoblock, and R. Van de Walle. Mapping hierarchical sources into rdf using the rml mapping language. In *Semantic Computing (ICSC), 2014 IEEE International Conference on*, pages 151–158. IEEE, 2014.

[90] M. C. Domingo. An overview of the internet of things for people with disabilities. *Journal of Network and Computer Applications*, 35(2):584–596, 2012.

[91] F. M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Reasoning in description logics. *Principles of knowledge representation*, 1:191–236, 1996.

[92] C. Doukas and I. Maglogiannis. Bringing iot and cloud computing towards pervasive healthcare. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pages 922–926. IEEE, 2012.

[93] S. Duquennoy, G. Grimaud, and J.-J. Vandewalle. Smews: Smart and mobile embedded web server. In *Complex, Intelligent and Software Intensive Systems, 2009. CISIS'09. International Conference on*, pages 571–576. IEEE, 2009.

[94] L. Erazo-Garzon, J. Avila, S. Pinos, and P. Cedillo. A systematic review on the use of ontologies in the internet of things. In *International Conference on Applied Technologies*, pages 509–524. Springer, 2021.

[95] B. A. Farshchian and T. Vilarinho. Which mobile health toolkit should a service provider choose? a comparative evaluation of apple healthkit, google fit, and samsung digital health platform. In *European Conference on Ambient Intelligence*, pages 152–158. Springer, 2017.

[96] L. Feigenbaum. Cambridge semantics position. In *W3C Workshop on RDF Next Steps, Stanford, Palo Alto, CA, USA*, 2010.

[97] A. Fensel, S. Tomic, V. Kumar, M. Stefanovic, S. V. Aleshin, and D. O. Novikov. Sesame-s: Semantic smart home system for energy efficiency. *Informatik-Spektrum*, 36(1):46–57, 2013.

[98] D. Fensel, F. Van Harmelen, I. Horrocks, D. L. McGuinness, and P. F. Patel-Schneider. Oil: An ontology infrastructure for the semantic web. *IEEE intelligent systems*, 16(2):38–45, 2001.

[99] F. Fernandez and G. C. Pallis. Opportunities and challenges of the internet of things for healthcare: Systems engineering perspective. In *Wireless Mobile Communication and Healthcare (Mobihealth), 2014 EAI 4th International Conference on*, pages 263–266. IEEE, 2014.

[100] T. M. Fernández-Caramés. An intelligent power outlet system for the smart home of the internet of things. *International Journal of Distributed Sensor Networks*, 11(11):214805, 2015.

[101] M. Fernández-López, A. Gómez-Pérez, and N. Juristo. Methontology: from ontological art towards ontological engineering. 1997.

[102] S. J. Fitz-Gerald and B. Wiggins. Staab, s., studer, r.(eds.), handbook on ontologies, series: International handbooks on information systems, vol. xix (2009). 811 p., 121 illus., hardcover£ 164, isbn: 978-3-540-70999-2. *International Journal of Information Management*, 30(1):98–100, 2010.

[103] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann. Modelling ontology evaluation and validation. In *European Semantic Web Conference*, pages 140–154. Springer, 2006.

[104] F. Ganz, D. Puschmann, P. Barnaghi, and F. Carrez. A practical evaluation of information processing and abstraction techniques for the internet of things. *IEEE Internet of Things journal*, 2(4):340–354, 2015.

[105] M. Ganzha, M. Paprzycki, W. Pawłowski, P. Szmeja, and K. Wasielewska. Semantic interoperability in the internet of things: an overview from the inter-iot perspective. *Journal of Network and Computer Applications*, 81:111–124, 2017.

[106] A. d. Garcez, M. Gori, L. C. Lamb, L. Serafini, M. Spranger, and S. N. Tran. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *arXiv preprint arXiv:1905.06088*, 2019.

[107] G. Garkoti, S. K. Peddoju, and R. Balasubramanian. Detection of insider attacks in cloud based e-healthcare environment. In *Information Technology (ICIT), 2014 International Conference on*, pages 195–200. IEEE, 2014.

[108] V. Gay and P. Leijdekkers. Bringing health and fitness data together for connected health care: mobile apps as enablers of interoperability. *Journal of medical Internet research*, 17(11), 2015.

[109] S. Goessner. Jsonpath (2007). *http://goessner.net/articles/JsonPath.*

[110] C. Golbreich, E. K. Wallace, and P. F. Patel-Schneider. Owl 2 web ontology language new features and rationale. *W3C working draft, W3C (June 2009) http://www.w3. org/TR/2009/WD-owl2-new-features-20090611*, 2009.

[111] C. Golbreich, S. Zhang, and O. Bodenreider. The foundational model of anatomy in owl: Experience and perspectives. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(3):181–195, 2006.

[112] J. Gold. Fda regulators face daunting task as health apps multiply. *USA Today*, 2012.

[113] A. Gomez-Perez, M. Fernández-López, and O. Corcho. *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web.* Springer Science & Business Media, 2006.

[114] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: combining logic programs with description logic. In *Proceedings of the 12th international conference on World Wide Web*, pages 48–57. ACM, 2003.

[115] W. O. W. Group. Owl 2 web ontology language. - w3c recommendation. https://www.w3.org/TR/owl2-overview/, 2012.

[116] W. O. W. Group. Sparql 1.1 overview. - w3c recommendation. https://www.w3.org/TR/sparql11-overview/, 2013.

[117] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.

[118] T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5-6):907–928, 1995.

[119] M. Grüninger and M. S. Fox. Methodology for the design and evaluation of ontologies. 1995.

[120] N. Guarino et al. Formal ontology and information systems. In *Proceedings of FOIS*, volume 98, pages 81–97, 1998.

[121] D. Guinard and V. Trifa. Towards the web of things: Web mashups for embedded devices. In *Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain*, volume 15, 2009.

[122] M. Haghi, K. Thurow, and R. Stoll. Wearable devices in medical internet of things: scientific research and commercially available devices. *Healthcare informatics research*, 23(1):4–15, 2017.

[123] A. Halevy, A. Doan, and Z. Ives. Principles of data integration. *Morgan Kaufmann*, 2012.

[124] A. Halevy, A. Doan, and Z. Ives. Current status and future trends in mobile health. *Research2Guidance*, mHealth App Economics 2017/2018.

[125] S. Handschuh and S. Staab. Annotation for the semantic web, frontiers in artificial intelligence and applications, vol. 96, 2003.

[126] D. Hardt. The oauth 2.0 authorization framework. Technical report, 2012.

[127] J. Hendler and D. L. McGuinness. The darpa agent markup language. *IEEE Intelligent systems*, 15(6):67–73, 2000.

[128] P. Heyvaert, A. Dimou, R. Verborgh, E. Mannens, and R. Van de Walle. Semantically annotating ceur-ws workshop proceedings with rml. In *Semantic Web Evaluation Challenge*, pages 165–176. Springer, 2015.

[129] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph. Owl 2 web ontology language primer. *W3C recommendation*, 27(1):123, 2009.

[130] 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*, 21:79, 2004.

[131] I. Horrocks, P. F. Patel-Schneider, and F. Van Harmelen. Reviewing the design of daml+ oil: An ontology language for the semantic web. *AAAI/IAAI*, 2002:792–797, 2002.

[132] I. Iakovidis. Towards personal health record: current situation, obstacles and trends in implementation of electronic healthcare record in europe. *International journal of medical informatics*, 52(1):105–115, 1998.

[133] E. T. S. Institute. Smartm2m; iot standards landscape and future evolutions. $https://aioti.eu/wp\text{-}content/uploads/2017/03/tr_103375v010101p - Standards - landscape - and - future - evolutions.pdf$, 2016.

[134] D. Isern, D. Sánchez, and A. Moreno. Ontology-driven execution of clinical guidelines. *Computer methods and programs in biomedicine*, 107(2):122–139, 2012.

[135] S. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak. The internet of things for health care: a comprehensive survey. *IEEE Access*, 3:678–708, 2015.

[136] R. S. Istepanian, S. Hu, N. Y. Philip, and A. Sungoor. The potential of internet of m-health things "m-iot" for non-invasive glucose level sensing. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 5264–5266. IEEE, 2011.

[137] M. Janssen, Y. Charalabidis, and A. Zuiderwijk. Benefits, adoption barriers and myths of open data and open government. *Information systems management*, 29(4):258–268, 2012.

[138] A. J. Jara, A. C. Olivieri, Y. Bocchi, M. Jung, W. Kastner, and A. F. Skarmeta. Semantic web of things: an analysis of the application semantics for the iot moving towards the iot convergence. *International Journal of Web and Grid Services*, 10(2-3):244–272, 2014.

[139] R. Jasper, M. Uschold, et al. A framework for understanding and classifying ontology applications. In *Proceedings 12th Int. Workshop on Knowledge Acquisition, Modelling, and Management KAW*, volume 99, pages 16–21, 1999.

[140] Z. Jian, W. Zhanli, and M. Zhuang. Temperature measurement system and method based on home gateway. *Chinese Patent*, 102(811):185, 2012.

[141] M. R. Kamdar and M. J. Wu. Prism: A data-driven platform for monitoring mental health. In *PSB*, pages 333–344, 2016.

[142] K. Kawamoto, T. Tanaka, and H. Kuriyama. Your activity tracker knows when you quit smoking. In *Proceedings of the 2014 ACM international symposium on wearable computers*, pages 107–110. ACM, 2014.

[143] H. H. Kim, S. Y. Lee, S. Y. Baik, and J. H. Kim. Mello: Medical lifelog ontology for data terms from self-tracking and lifelog devices. *International journal of medical informatics*, 84(12):1099–1110, 2015.

[144] A. Kiryakov, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff. Semantic annotation, indexing, and retrieval. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2(1):49–79, 2004.

[145] P. Klasnja and W. Pratt. Healthcare in the pocket: mapping the space of mobile-phone health interventions. *Journal of biomedical informatics*, 45(1):184–198, 2012.

[146] N. Konstantinou and D.-E. Spanos. *Materializing the Web of Linked Data*. Springer, 2015.

[147] C. E. Koop, R. Mosher, L. Kun, J. Geiling, E. Grigg, S. Long, C. Macedonia, R. C. Merrell, R. Satava, and J. M. Rosen. Future delivery of health care: Cybercare. *IEEE Engineering in Medicine and Biology Magazine*, 27(6), 2008.

[148] G. Kopanitsa, C. Hildebrand, J. Stausberg, and K. Englmeier. Visualization of medical data based on ehr standards. *Methods of information in medicine*, 52(01):43–50, 2013.

[149] L. Lacy, G. Aviles, K. Fraser, W. Gerber, A. M. Mulvehill, and R. Gaskill. Experiences using owl in military applications. In *OWLED*, volume 188, 2005.

[150] A. A. Laghari, K. Wu, R. A. Laghari, M. Ali, and A. A. Khan. A review and state of art of internet of things (iot). *Archives of Computational Methods in Engineering*, pages 1–19, 2021.

[151] G. Lakoff. *Women, fire, and dangerous things*. University of Chicago press, 2008.

[152] J.-B. Lamy. Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies. *Artificial intelligence in medicine*, 80:11–28, 2017.

[153] E. C. Larson, M. Goel, G. Boriello, S. Heltshe, M. Rosenfeld, and S. N. Patel. Spirosmart: using a microphone to measure lung function on a mobile phone. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 280–289. ACM, 2012.

[154] J. Lee, B. A. Reyes, D. D. McManus, O. Maitas, and K. H. Chon. Atrial fibrillation detection using an iphone 4s. *IEEE Transactions on Biomedical Engineering*, 60(1):203–206, 2013.

[155] K.-M. Lee, W.-G. Teng, and T.-W. Hou. Point-n-press: An intelligent universal remote control system for home appliances. *IEEE Transactions on automation science and engineering*, 13(3):1308–1317, 2016.

[156] M. Lefrançois, A. Zimmermann, and N. Bakerally. A sparql extension for generating rdf from heterogeneous formats. In *European Semantic Web Conference*, pages 35–50. Springer, 2017.

[157] N. Lopes, A. Zimmermann, A. Hogan, G. Lukácsy, A. Polleres, U. Straccia, and S. Decker. Rdf needs annotations. In *W3C Workshop on RDF Next Steps, Stanford, Palo Alto, CA, USA*, 2010.

[158] W. Ludwig, K.-H. Wolf, C. Duwenkamp, N. Gusew, N. Hellrung, M. Marschollek, M. Wagner, and R. Haux. Health-enabling technologies for the elderly–an overview of services based on a literature

review. *Computer methods and programs in biomedicine*, 106(2):70–78, 2012.

[159] Y.-J. Ma, Y. Zhang, O. M. Dung, R. Li, and D.-Q. Zhang. Health internet of things: recent applications and outlook. *Journal of Internet Technology*, 16(2):351–362, 2015.

[160] L. M. O. Machado, R. R. Souza, and M. da Graça Simões. Semantic web or web of data? a diachronic study (1999 to 2017) of the publications of tim berners-lee. 2018.

[161] A. Madan, S. T. Moturu, D. Lazer, and A. S. Pentland. Social sensing: obesity, unhealthy eating and exercise in face-to-face networks. In *Wireless Health 2010*, pages 104–110. ACM, 2010.

[162] M. L. Maliniak, A. V. Patel, M. L. McCullough, P. T. Campbell, C. R. Leach, S. M. Gapstur, and M. M. Gaudet. Obesity, physical activity, and breast cancer survival among older breast cancer survivors in the cancer prevention study-ii nutrition cohort. *Breast cancer research and treatment*, 167(1):133–145, 2018.

[163] F. Manola, E. Miller, B. McBride, et al. Rdf primer. *W3C recommendation*, 10(1-107):6, 2004.

[164] J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, and A. Marrs. Disruptive technologies: Advances that will transform life, business, and the global economy (vol. 12): Mckinsey global institute san francisco, ca, 2013.

[165] V. Marinakis and H. Doukas. An advanced iot-based system for intelligent energy management in buildings. *Sensors*, 18(2):610, 2018.

[166] M. S. Marshall, R. Boyce, H. F. Deus, J. Zhao, E. L. Willighagen, M. Samwald, E. Pichler, J. Hajagos, E. Prudâhommeaux, and S. Stephens. Emerging practices for mapping and linking life sciences

data using rdfâa case series. *Web Semantics: Science, Services and Agents on the World Wide Web*, 14:2–13, 2012.

[167] L. Masinter, T. Berners-Lee, and R. T. Fielding. Uniform resource identifier (uri): Generic syntax. 2005.

[168] K. Matsumura and T. Yamakoshi. iphysiometer: a new approach for measuring heart rate and normalized pulse volume using only a smartphone. *Behavior research methods*, 45(4):1272–1278, 2013.

[169] D. L. McGuinness. *Ontologies come of age.* Mit Press, 2005.

[170] D. L. McGuinness, R. Fikes, J. Hendler, and L. A. Stein. Daml+ oil: an ontology language for the semantic web. *IEEE Intelligent Systems*, 17(5):72–80, 2002.

[171] D. L. McGuinness, F. Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

[172] X. Mi, F. Qian, Y. Zhang, and X. Wang. An empirical characterization of ifttt: ecosystem, usage, and performance. In *Proceedings of the 2017 Internet Measurement Conference*, pages 398–404, 2017.

[173] R. V. Milani and C. J. Lavie. Health care 2020: reengineering health care delivery to combat chronic disease. *The American journal of medicine*, 128(4):337–343, 2015.

[174] E. Miller. An introduction to the resource description framework. *Bulletin of the Association for Information Science and Technology*, 25(1):15–19, 1998.

[175] G. Miller. Wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–312, 1990.

[176] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[177] L. W. Min, G. Wei, C. Yeshen, H. Yannian, W. Yiying, and Z. S. Home. Architecture, technologies and systems. In *proceedings of 8th International Congress of Information and Communication Technology (ICICT-2018). ScienceDirect Procedia Computer Science*, volume 131, 2018.

[178] J. Moreira, L. F. Pires, M. van SINDEREN, and L. Daniele. Saref4health: Iot standard-based ontology-driven healthcare systems. In *FOIS*, pages 239–252, 2018.

[179] A. S. M. Mosa, I. Yoo, and L. Sheets. A systematic review of healthcare applications for smartphones. *BMC medical informatics and decision making*, 12(1):67, 2012.

[180] B. Motik, P. F. Patel-Schneider, and B. C. Grau. Owl 2 web ontology language direct semantics. *W3C recommendation*, 27, 2009.

[181] B. Motik, U. Sattler, and R. Studer. Query answering for owl-dl with rules. *Journal of Web Semantics*, 3(1):41–60, 2005.

[182] S. T. Moturu, I. Khayal, N. Aharony, W. Pan, and A. Pentland. Sleep, mood and sociability in a healthy population. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 5267–5270. IEEE, 2011.

[183] N. Z. Naqvib, A. Kumar, A. Chauhan, and K. Sahni. Step counting using smartphone-based accelerometer. *International Journal on Computer Science and Engineering*, 4(5):675, 2012.

[184] T. H. Nelson. Complex information processing: a file structure for the complex, the changing and the indeterminate. In *Proceedings of the 1965 20th national conference*, pages 84–100. ACM, 1965.

[185] N. Noy, A. Rector, P. Hayes, and C. Welty. Defining n-ary relations on the semantic web. *W3C working group note*, 12(4), 2006.

[186] N. F. Noy, D. L. McGuinness, et al. Ontology development 101: A guide to creating your first ontology, 2001.

[187] N. F. Noy, M. Sintek, S. Decker, M. Crubézy, R. W. Fergerson, and M. A. Musen. Creating semantic web contents with protege-2000. *IEEE intelligent systems*, 16(2):60–71, 2001.

[188] L. Obrst, W. Ceusters, I. Mani, S. Ray, and B. Smith. The evaluation of ontologies. In *Semantic web*, pages 139–158. Springer, 2007.

[189] OpenWeatherMap.org. ¡urrent weather and forecast.

[190] I. Pagkalos and L. Petrou. Senhance: A semantic web framework for integrating social and hardware sensors in e-health. *Health informatics journal*, 22(3):505–522, 2016.

[191] Z. Pang. *Technologies and Architectures of the Internet-of-Things (IoT) for Health and Well-being*. PhD thesis, KTH Royal Institute of Technology, 2013.

[192] P. Pediaditis, G. Flouris, I. Fundulaki, and V. Christophides. On explicit provenance management in rdf/s graphs. In *Workshop on the Theory and Practice of Provenance*, 2009.

[193] D. Pekmezi, C. Ainsworth, T. Holly, V. Williams, T. Benitez, K. Wang, L. Q. Rogers, B. Marcus, and W. Demark-Wahnefried. Rationale, design, and baseline findings from a pilot randomized trial of an ivr-supported physical activity intervention for cancer prevention in the deep south: The dial study. *Contemporary clinical trials communications*, 8:218–226, 2017.

[194] D. Pfisterer, K. Romer, D. Bimschas, O. Kleine, R. Mietz, C. Truong, H. Hasemann, A. Kröller, M. Pagel, M. Hauswirth, et al. Spitfire: toward a semantic web of things. *IEEE Communications Magazine*, 49(11):40–48, 2011.

[195] K. T. Pickard and M. Swan. Big desire to share big health data: A shift in consumer attitudes toward personal health information. In *2014 AAAI Spring Symposium Series*, pages 2168–7161, 2014.

[196] I. Plaza, L. MartíN, S. Martin, and C. Medrano. Mobile applications in an aging society: Status and trends. *Journal of Systems and Software*, 84(11):1977–1988, 2011.

[197] M.-Z. Poh, D. J. McDuff, and R. W. Picard. Advancements in noncontact, multiparameter physiological measurements using a webcam. *IEEE transactions on biomedical engineering*, 58(1):7–11, 2011.

[198] M. Poveda-Villalón, A. Gómez-Pérez, and M. C. Suárez-Figueroa. Oops!(ontology pitfall scanner!): An on-line tool for ontology evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):7–34, 2014.

[199] L. Prieto Gonzalez, A. Fensel, J. M. Gomez Berbis, A. Popa, and A. de Amescua Seco. A survey on energy efficiency in smart homes and smart grids. *Energies*, 14(21), 2021.

[200] E. Prud, A. Seaborne, et al. Sparql query language for rdf. 2006.

[201] J. Puustjärvi and L. Puustjärvi. The role of smart data in smart home: health monitoring case. *Procedia Computer Science*, 69:143–151, 2015.

[202] H. Rahman and M. I. Hussain. A comprehensive survey on semantic interoperability for internet of things: State-of-the-art and research challenges. *Transactions on Emerging Telecommunications Technologies*, 31(12):e3902, 2020.

[203] P. R. M. Rao, S. M. Krishna, and A. S. Kumar. Privacy preservation techniques in big data analytics: a survey. *Journal of Big Data*, 5(1):33, 2018.

[204] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity recognition from accelerometer data. In *Aaai*, volume 5, pages 1541–1546, 2005.

[205] R. Rawassizadeh, B. A. Price, and M. Petre. Wearables: Has the age of smartwatches finally arrived? *Communications of the ACM*, 58(1):45–47, 2015.

[206] R. Reda, F. Piccinini, and A. Carbonaro. Semantic modelling of smart healthcare data. In *Proceedings of SAI Intelligent Systems Conference*, pages 399–411. Springer, 2018.

[207] R. Reda, F. Piccinini, and A. Carbonaro. Towards consistent data representation in the iot healthcare landscape. In *Proceedings of the 2018 International Conference on Digital Health*, pages 5–10. ACM, 2018.

[208] R. Reda, F. Piccinini, G. Martinelli, and A. Carbonaro. Heterogeneous self-tracked health and fitness data integration and sharing according to a linked open data approach. *Computing*, pages 1–23, 2021.

[209] B. Reeder and A. David. Health at hand: a systematic review of smart watch uses for health and wellness. *Journal of biomedical informatics*, 63:269–276, 2016.

[210] A. Rhayem, M. B. A. Mhiri, M. B. Salah, and F. Gargouri. Ontology-based system for patient monitoring with connected objects. *Procedia Computer Science*, 112:683–692, 2017.

[211] C. Richard, W. David, and L. Markus. Rdf 1.1 concepts and abstract syntax - w3c recommendation. https://www.w3.org/TR/rdf11-concepts/, 2014.

[212] I. Richard. An introduction to multilingual web addresses. https://www.w3.org/International/articles/idn-and-iri/, 2008.

[213] D. Roman, N. Nikolov, A. Putlier, D. Sukhobok, B. Elvesæter, A. Berre, X. Ye, M. Dimitrov, A. Simov, M. Zarev, et al. Datagraft: One-stop-shop for open data management. *Semantic Web*, 9(4):393–411, 2018.

[214] M. Ruckenstein and M. Pantzar. Beyond the quantified self: Thematic exploration of a dataistic paradigm. *New Media & Society*, 19(3):401–418, 2017.

[215] M. Ruiz, J. García, and B. Fernández. Body temperature and its importance as a vital constant. *Revista de enfermeria (Barcelona, Spain)*, 32(9):44–52, 2009.

[216] D. Saba, O. Cheikhrouhou, W. Alhakami, Y. Sahli, A. Hadidi, and H. Hamam. Intelligent reasoning rules for home energy management (irrhem): Algeria case study. *Applied Sciences*, 12(4):1861, 2022.

[217] D. Saba, Y. Sahli, and A. Hadidi. An ontology based energy management for smart home. *Sustainable Computing: Informatics and Systems*, 31:100591, 2021.

[218] M. Salvadores, P. R. Alexander, M. A. Musen, and N. F. Noy. Bioportal as a dataset of linked biomedical ontologies and terminologies in rdf. *Semantic web*, 4(3):277–284, 2013.

[219] S. S. I. Samuel. A review of connectivity challenges in iot-smart home. In *2016 3rd MEC International conference on big data and smart city (ICBDSC)*, pages 1–4. IEEE, 2016.

[220] G. Santos, T. Pinto, Z. Vale, R. Carvalho, B. Teixeira, and C. Ramos. Upgrading bricksâthe context-aware semantic rule-based system for intelligent building energy and security management. *Energies*, 14(15):4541, 2021.

[221] G. Santucci et al. From internet of data to internet of things. In *International Conference on Future Trends of the Internet*, volume 28, 2009.

[222] S. Sepasgozar, R. Karimi, L. Farahzadi, F. Moezzi, S. Shirowzhan, S. M Ebrahimzadeh, F. Hui, and L. Aye. A systematic content review of artificial intelligence and the internet of things applications in smart home. *Applied Sciences*, 10(9):3074, 2020.

[223] R. Shearer, B. Motik, and I. Horrocks. Hermit: A highly-efficient owl reasoner. In *OWLED*, volume 432, page 91, 2008.

[224] F. Shen and Y. Lee. Biobroker: Knowledge discovery framework for heterogeneous biomedical ontologies and data. *Journal of Intelligent Learning Systems and Applications*, 10(01):1, 2018.

[225] R. J. Shephard. Physical activity and prostate cancer: an updated review. *Sports Medicine*, 47(6):1055–1073, 2017.

[226] A. Sheth. Internet of things to smart iot through semantic, cognitive, and perceptual computing. *IEEE Intelligent Systems*, 31(2):108–112, 2016.

[227] A. Sheth, P. Anantharam, and C. Henson. Semantic, cognitive, and perceptual computing: Paradigms that shape human experience. *Computer*, 49(3):64–72, 2016.

[228] A. Sidhu, T. S. Dillon, E. Chang, and B. Sidhu. Protein ontology development using owl. In *Proceedings of the 2005 Workshop on OWL: Experiences and Directions (OWLED'05)*. CEUR Workshop Proceedings, 2005.

[229] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical owl-dl reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 5(2):51–53, 2007.

[230] C. Snomed. Systematized nomenclature of medicine-clinical terms. *International Health Terminology Standards Development Organisation*, 2011.

[231] D. Soergel, B. Lauser, A. Liang, F. Fisseha, J. Keizer, and S. Katz. Reengineering thesauri for new applications: the agrovoc example. *Journal of digital information*, 4(4), 2006.

[232] S. Sonune, D. Kalbande, A. Yeole, and S. Oak. Issues in iot healthcare platforms: A critical study and review. In *2017 International Conference on Intelligent Computing and Control (I2C2)*, pages 1–5. IEEE, 2017.

[233] J. F. Sowa et al. *Knowledge representation: logical, philosophical, and computational foundations*, volume 13. MIT Press, 2000.

[234] S. Staab, R. Studer, H.-P. Schnurr, and Y. Sure. Knowledge processes and ontologies. *IEEE Intelligent systems*, 16(1):26–34, 2001.

[235] D. M. Steinberg, D. F. Tate, G. G. Bennett, S. Ennett, C. Samuel-Hodge, and D. S. Ward. The efficacy of a daily self-weighing weight loss intervention using smart scales and e-mail. *Obesity*, 21(9):1789–1797, 2013.

[236] C. P. Stone. A glimpse at ehr implementation around the world: The lessons the us can learn. *Health Institute for E-Health Policy, May*, 2014.

[237] X. Su, H. Tong, and P. Ji. Activity recognition with smartphone sensors. *Tsinghua Science and Technology*, 19(3):235–249, 2014.

[238] M. Swan. Sensor mania! the internet of things, wearable computing, objective metrics, and the quantified self 2.0. *Journal of Sensor and Actuator Networks*, 1(3):217–253, 2012.

[239] L. Sweeney, A. Abu, and J. Winn. Identifying participants in the personal genome project by name (a re-identification experiment). *arXiv preprint arXiv:1304.7605*, 2013.

[240] S. Taj, U. Asad, M. Azhar, and S. Kausar. Interoperability in iot based smart home: A review. *Journal homepage: http://iieta. org/Journals/RCES*, 5(3):50–55, 2018.

[241] M. Tao, K. Ota, and M. Dong. Ontology-based data semantic management and application in iot-and cloud-enabled smart homes. *Future generation computer systems*, 76:528–539, 2017.

[242] H. J. ter Horst. Completeness, decidability and complexity of entailment for rdf schema and a semantic extension involving the owl vocabulary. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(2):79–115, 2005.

[243] H. Tetteh. Method and system for monitoring congestive heart failure risk of a cardiac patient, Nov. 26 2015. US Patent App. 14/717,329.

[244] B. Tilahun, T. Kauppinen, C. Keßler, and F. Fritz. Design and development of a linked open data-based health information representation and visualization system: potentials and preliminary evaluation. *JMIR medical informatics*, 2(2), 2014.

[245] K. Tran, T. Le, and T. Dinh. A high-accuracy step counting algorithm for iphones using accelerometer. In *Signal Processing and Information Technology (ISSPIT), 2012 IEEE International Symposium on*, pages 000213–000217. IEEE, 2012.

[246] D. Tsarkov and I. Horrocks. Fact++ description logic reasoner: System description. *Automated reasoning*, pages 292–297, 2006.

[247] K. Tsubouchi, R. Kawajiri, and M. Shimosaka. Working-relationship detection from fitbit sensor data. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*, pages 115–118. ACM, 2013.

[248] S. Y. Y. Tun, S. Madanian, and F. Mirza. Internet of things (iot) applications for elderly care: a reflective review. *Aging clinical and experimental research*, 33(4):855–867, 2021.

[249] M. Uschold and M. Gruninger. Ontologies: Principles, methods and applications. *The knowledge engineering review*, 11(2):93–136, 1996.

[250] F. Van den Abeele, J. Hoebeke, I. Moerman, and P. Demeester. Integration of heterogeneous devices and communication models via the cloud in the constrained internet of things. *International Journal of Distributed Sensor Networks*, 11(10):683425, 2015.

[251] R. van der Weerdt, V. de Boer, L. Daniele, and B. Nouwt. Validating saref in a smart home environment. In *Research Conference on Metadata and Semantics Research*, pages 35–46. Springer, 2020.

[252] J.-P. Vasseur and A. Dunkels. *Interconnecting smart objects with ip: The next internet.* Morgan Kaufmann, 2010.

[253] T. Wadhawan, N. Situ, H. Rui, K. Lancaster, X. Yuan, and G. Zouridakis. Implementation of the 7-point checklist for melanoma detection on smart handheld devices. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 3180–3183. IEEE, 2011.

[254] L. Wei, Y. Heng, and W. Y. Lin. Things based wireless data transmission of blood glucose measuring instruments. *Chinese Patent*, 202(154):684, 2012.

[255] C. Welty, D. L. McGuinness, and M. K. Smith. Owl web ontology language guide. *W3C recommendation, W3C (February 2004) http://www. w3. org/TR/2004/REC-owl-guide-20040210*, 2004.

[256] P. L. Whetzel, N. F. Noy, N. H. Shah, P. R. Alexander, C. Nyulas, T. Tudorache, and M. A. Musen. Bioportal: enhanced functionality via new web services from the national center for biomedical ontology to access and use ontologies in software applications. *Nucleic acids research*, 39(suppl_2):W541–W545, 2011.

[257] G. Witmer. Dictionary of philosophy of mind- ontology. *Retrieved May*, 11:2004, 2004.

[258] L. Wood, A. Le Hors, V. Apparao, S. Byrne, M. Champion, S. Isaacs, I. Jacobs, G. Nicol, J. Robie, R. Sutor, et al. Document object model (dom) level 1 specification. *W3C Recommendation*, 1, 1998.

[259] C.-L. Wu, Y.-S. Tseng, and L.-C. Fu. Spatio-temporal feature enhanced semi-supervised adaptation for activity recognition in iot-based context-aware smart homes. In *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, pages 460–467. IEEE, 2013.

[260] D. Zeng, S. Guo, and Z. Cheng. The web of things: A survey. *Journal of Communications*, 6(6):424–438, 2011.

[261] X. Zenuni, B. Raufi, F. Ismaili, and J. Ajdari. State of the art of semantic web for healthcare. *Procedia-Social and Behavioral Sciences*, 195:1990–1998, 2015.

[262] Z. Zhai, J.-F. Martínez Ortega, N. Lucas Martínez, and P. Castillejo. A rule-based reasoner for underwater robots using owl and swrl. *Sensors*, 18(10):3481, 2018.