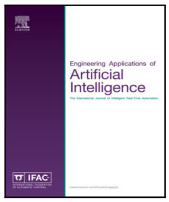
Engineering Applications of Artificial Intelligence 97 (2021) 104026

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Conformance testing of ontologies through ontology requirements

Alba Fernández-Izquierdo *, Raúl García-Castro

Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

ARTICLE INFO

Keywords:
Ontology conformance
Ontology engineering
Ontology testing
Standard

ABSTRACT

In recent years, several standard ontologies have been developed to maximise semantic interoperability in different domains; such standard ontologies ensure quality and integrity when describing a domain. Therefore, mechanisms to guarantee that developers build ontologies that conform to such standards are needed. However, while in fields such as Software Engineering or industry, conformance testing plays an essential role during product development, in the Ontology Engineering field there is a lack of techniques for this type of testing. This work introduces an ontology conformance testing method to analyse conformance between an ontology and a standard based on the standard requirements. Grounded on this method, the work also presents a minimum common knowledge identification method for analysing how a group of standards covers a particular domain and for identifying whether there are conflicts between them. This work has been validated by analysing the conformance between an ontology network and a set of standards on the Internet of Things domain, and by analysing the minimum common knowledge between such standards. This analysis shows that the conformance between ontologies and standards is mostly related to definition of classes. Furthermore, the analysis shows that although the analysed standards are related to the same domain, they are created to describe different areas of concern and, thus, there is a minimum overlap between them. Finally, it was concluded that the quality of the conformance analysis depends on the quality of the requirements specification: the more precise the requirements, the more precise the analysis between ontologies and standards.

1. Introduction

In the Software Engineering field, conformance testing refers to techniques that determine to what extent an implementation of a particular standard conforms to the requirements of that standard (Moseley et al., 2003). Ensuring conformance plays an important role in systems across many domains and serves to ensure that a product, process, computer program or system meets a defined set of standards and thus is reliable and interoperable (Graydon et al., 2012).

Currently, several standardisation bodies are generating standard ontologies, i.e., ontological specifications that are published as a standard or as a part of one, to maximise semantic interoperability. A standard provides an agreed baseline approved by the community that can be used by other developers who want to describe the related area of concern. This standard can be used by developers to identify potential areas of conflict between the standard and their ontology as the result of embedded differences when describing the domain. Therefore, it should be possible to ensure that the ontologies of a particular domain conform to a particular standard, ensuring quality and interoperability when describing such domain. This analysis of conformance would reflect the absence of inconsistencies and the presence of overlaps according to the ontologies and the standard specification. As examples of standard specifications, on the Internet of Things (IoT) domain, several standard ontologies have been defined: the European Telecommunications Standards Institute (ETSI) developed the SAREF (ETSI, 2017) ontology for describing smart applications, the World Wide Web Consortium (W3C) developed the SSN (Haller et al., 2019) ontology for describing sensors and their observations, and the oneM2M organisation developed the oneM2M Base ontology (oneM2M, 2016) for providing syntactic and semantic interoperability of oneM2M platforms with external systems.

Until now, conformance analyses in the Semantic Web field were focused on evaluating conformance between specifications of semantic technologies and ontology representation languages, e.g., regarding the OWL language (García-Castro, 2009) or regarding SPARQL (Prud'hommeaux and Seaborne, 2008), but no specific work has been performed in the Ontology Engineering field to analyse conformance according to an ontology specification, e.g., its associated requirements. Therefore, formal methods to automate the conformance analysis between an ontology and a standard specification are needed. The first contribution of this paper is a conformance testing method for ontologies based on requirements inspired by the ISO/IEC 9646 conformance norm (ISO/IEC 9646-1, 1994), which is a conformance standard used in Software Engineering.

* Corresponding author.
E-mail addresses: albafernandez@fi.upm.es (A. Fernández-Izquierdo), rgarcia@fi.upm.es (R. García-Castro).

https://doi.org/10.1016/j.engappai.2020.104026
Received 16 April 2020; Received in revised form 18 September 2020; Accepted 15 October 2020
Available online 22 October 2020

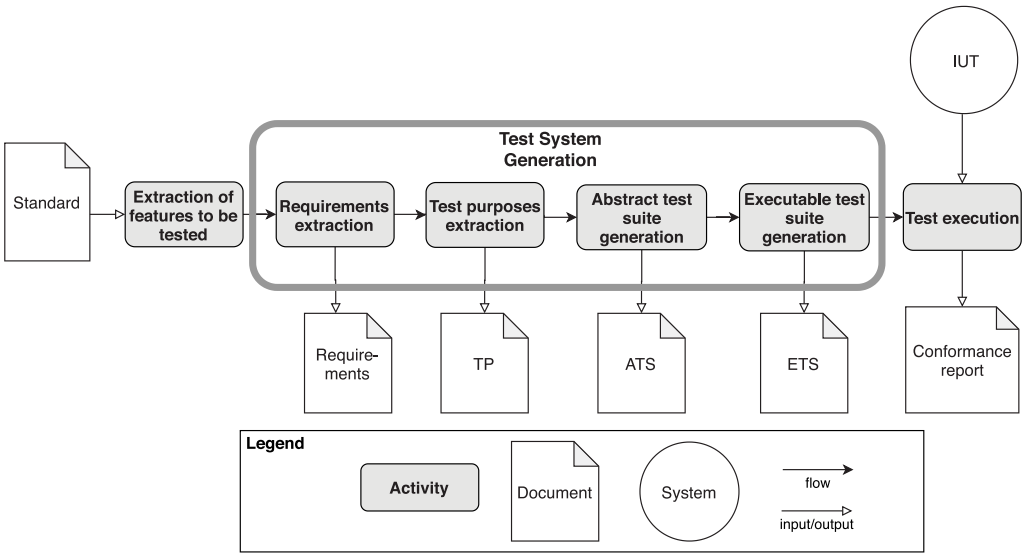


Fig. 1. Summary of ISO/IEC 9646 activities for conformance testing.

To define this method, the translation between requirements from standards into tests was analysed to generate tests from a standard specification. These standard tests must be executed on ontologies that are not related to the standard specification. Therefore, such tests must be reusable. Moreover, it was also analysed which is the information related to the conformance testing process that must be included in the conformance report, allowing to identify the inconsistencies and overlaps found between an ontology and the standard.

Intending to analyse the overlap between a set of standards in a particular domain, this work also presents a method for identifying the minimum common knowledge between ontologies. Therefore, it enables to determine to what extent a set of standard ontologies represents the same knowledge and which are their differences.

This approach for analysing ontology conformance through ontological requirements aims at addressing the following research questions to be aware about the shared knowledge between ontologies related to the same domain:

RQ1. *What is the level of overlap in terms of requirements between an ontology and a standard from the same domain?*
RQ2. *Which types of requirements support interoperability, i.e., are shared by ontologies from the same domain?*

The remainder of this paper is structured as follows. Section 2 presents the state of the art related to existing conformance testing methods. Section 3 describes the proposed method for conformance testing in Ontology Engineering. Section 4 describes the method to analyse the minimum common knowledge between ontologies, while Section 5 describes validation of the proposed research questions. Finally, Section 6 outlines some conclusions and future work.

2. State of the art

This section presents the most well-known conformance testing approaches in the Software Engineering field, which are used as the basis for the work presented in this paper. Moreover, although in the Ontology Engineering field there is a lack of approaches for analysing conformance between ontologies and standards, this section also presents the most relevant ontology testing approaches, which can be considered as the basis of any ontology conformance method.

2.1. Conformance testing methods

The following sub-sections present a summary of the most relevant conformance testing approaches for industry and software, which are supported by several standardisation bodies and are used as the basis of this work. This section presents the norm for conformance testing as published by the International Organisation for Standardisation (ISO) and the International Electrotechnical Commission (IEC), as well as the conformance approaches developed by the ETSI and the W3C.

2.1.1. Conformance testing by the ISO/IEC

ISO/IEC 9646 (ISO/IEC 9646-1, 1994) is a multi-part international standard which specifies a general methodology for testing the conformance of products to Open Systems Interconnection specifications. However, the concepts for testing their implementations have a broader applicability and can also be used in testing of other kinds of protocol systems. The standard also defines TTCN (Kristoffersen and Walter, 1996), a tree and tabular notation for a formal description of test cases whose language is independent of technology, operating system and implementation domain.

The process of conformance testing in this ISO/IEC norm begins with the collation and categorisation of the features and options to be tested into a tabular form which is normally referred to as the implementation conformance statement (ICS). All implemented capabilities supported by the implementation under test are listed by the implementer in the ICS, so that the tester knows which options have to be tested.

The next step is to collect the requirements. For each requirement, one or more tests should be identified and expressed in the form of test purposes (TP), which describe a well-defined objective of testing. The TP describe in plain language the actions required to reach a verdict on whether an implementation passes or fails the test. Then, the tests are classified into a number of groups to provide the test suite structure (TSS). The test cases are combined into an abstract test suite (ATS) using a specific testing language such as TTCN. It is worth mentioning that the test suite is abstract in the sense that the tests are developed independently of any implementation.

Based on the ATS, a set of executable test cases (ETS) is generated. Such ETSs are then verified against a number of implementations to test (IUT) for correct operation according to some agreed procedures. An implementation extra information for test (IXIT) associated to the ATS should be produced to help executing protocol conformance testing. The results of this verification process are documented in a conformance report. Fig. 1 summarises these activities.

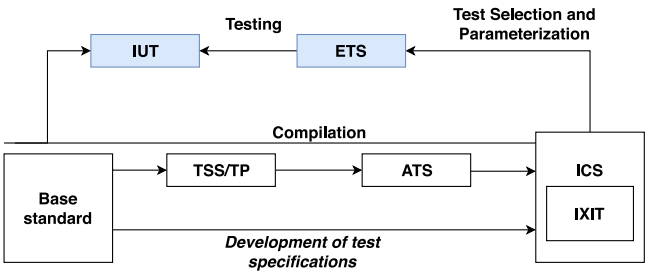


Fig. 2. Development of ETSI test specifications (Moseley et al., 2003).

2.1.2. Conformance testing by the ETSI

The ETSI conformance testing specifications are developed according to the method described in the ISO/IEC 9646. The specification provided by the ETSI (Moseley et al., 2003) is also focused on several artefacts, previously defined in the ISO/IEC 9646, namely, the Implementation Conformance Statement (ICS), the Implementation eXtra Information for Testing (IXIT), the Testing Purposes (TP), the Abstract Test Suite (ATS), and the Executable Test Suite (ETS). Fig. 2 shows the process development of test specifications.

Similarly as in the ISO/IEC 9646, the ATS represents the entire collection of test cases. In the ETSI conformance testing method, each test case specifies the detailed coding of the TP written using the standardised test specification language TTCN-3 (Grabowski et al., 2003), which is an updated version of TTCN. The ETS can be implemented from the ATS using the TTCN compilers available on testing platforms.

All the test specifications developed by ETSI are available online and can be searched and downloaded via the ETSI Work Programme application.[1] These specifications include a description of the tests in a human-readable format as well as the executable tests written following the TTCN language.

2.1.3. Conformance testing by the W3C

The conformance testing approach presented by the W3C is focused on the requirements and good practices for including conformance in W3C specifications (Dubost et al., 2005). It emphasises the need of conformance clauses in order to develop successful interoperability of implementations, rather than on defining an homogeneous way to define tests for analysing conformance.

The W3C proposes conformance requirements such as the addition of conformance clauses in every specification and the identification of which conformance requirements are mandatory, recommended and optional. Additionally, it also requires to use a consistent style for conformance requirements and to explain how to distinguish them. An example of detailed conformance clauses following the W3C guidelines is included in the Scalable Vector Graphics 1.1 specification (Dahlström et al., 2011), which describes all the requirements that should be fulfilled.

Regarding good practices, the W3C proposes the definition of the specification conformance model and the specification of how to distinguish normative from informative content, as well as the description of the wording for conformance claims.

2.2. Ontology testing methods

In the Ontology Engineering field, there is active research on ontology evaluation (Duque-Ramos et al., 2011; Lozano-Tello and Gómez-Pérez, 2004; Poveda-Villalón et al., 2014; Guarino and Welty, 2002; Suárez-Figueroa et al., 2012). However, after analysing the literature looking for conformance approaches in this field, i.e., approaches for ensuring that an ontology satisfies the specification of a standard, it was concluded that in the state of the art there are no works which deal with the conformance perspective as it is presented in Section 2.1. Some initiatives that deal with ontology testing from requirements were found, which could be considered as the first step in a conformance analysis. Such approaches are described in the following sub-sections.

2.2.1. Blomqvist and colleagues' testing approach

Blomqvist and colleagues (Blomqvist et al., 2012) presented an agile approach which includes a methodological background for testing and introduces three main types of tests based on the purpose of the test, namely, competency question verification, inference verification, and error provocation. The first type of test is oriented to the reformulation of the competency questions as SPARQL queries after adding test data related to the query to be reformulated, i.e., it does check classes and properties in the ontology. The second type verifies that the inference mechanisms are in place. Finally, the third one is intended to expose faults.

In order to keep tests separated from the ontology to be tested, this proposal represents a test case as an OWL ontology, which includes properties for describing each test case.

Although this methodology proposes different types of tests to verify requirements, which could be considered as a set of TPs, it does not describe the relation between such requirements and the tests. The tests in this approach are not abstract and, therefore, they cannot be executed on several ontologies, because the SPARQL query is not independent of the ontology from which it is extracted.

2.2.2. CQChecker

CQChecker[2] (Bezerra et al., 2014) is a tool proposed by Bezerra and colleagues which provides a mechanism to verify whether an ontology meets a set of competency questions by supporting both assertional and terminological queries. To accomplish that, the authors distinguish several types of competency questions such as competency questions which work over classes and their relations or competency questions which work over instances.

Unlike Blomqvist and colleagues, which proposed a set of the generic types of tests, the authors identified three types of requirements written as competency questions to be verified based on how they are specified:

- Competency questions which work over classes and their relations.
- Decision problems expressed as competency questions. In this type, the answer permitted to the question can only be true or false.
- Competency questions are expressed in an interrogative form which works only over instances.

This tool allows translating these three types of competency questions, which represent the ontology requirements, into SPARQL queries and executing them on an ontology. The set of SPARQL queries could represent a set of TPs associated to an ontology. Moreover, since the translation between the requirement and the SPARQL query is automatic depending on the ontology in which the SPARQL query is executed, the set of TPs can be considered abstract. However, authors are focused on the technological support to execute such tests rather on the methodology for performing the testing process. Therefore, there is a lack of a formal process to generate the tests and to analyse the obtained results.

[1] https://portal.etsi.org/webapp/WorkProgram/SimpleSearch/QueryForm.asp.

[2] CQChecker is available in the following URL: https://sourceforge.net/projects/cqchecker. The last update was on December 5, 2016.

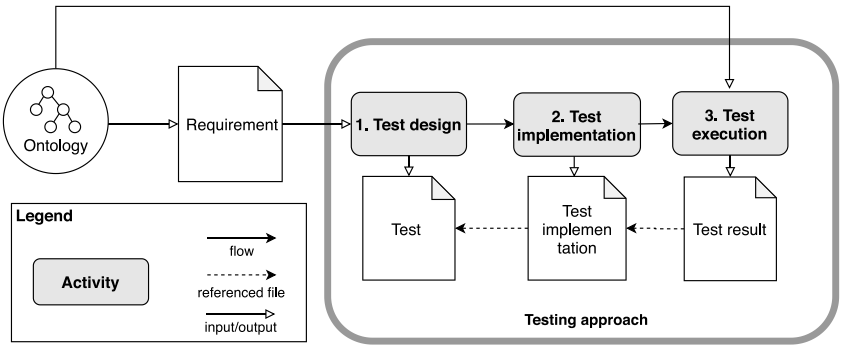


Fig. 3. Testing activities during the testing process, together with their inputs and outputs (Fernández-Izquierdo and García-Castro, 2018).

2.2.3. Test-driven development of ontologies

Keet and Ławrynowicz proposed a test-driven development (TDD) of ontologies (Keet and Ławrynowicz, 2016), which is an ontology development method inspired by the test-driven development approach in Software Engineering (Beck, 2003). This TDD, which is based on the idea of writing a failing test before writing any code, ensures that what is added to the ontology does have the intended effect specified upfront.

The TDD method is supported by the TDDOnto tool,[3] which has been first introduced by Lawrynowicz and Keet (2016) and further developed as TDDOnto2 (Davies et al., 2017). This tool checks whether a particular axiom is present in an ontology by using OWL Manchester Syntax axioms (Horridge and Patel-Schneider, 2012) as tests.

This approach is oriented to the definition of a set of TPs and their execution on an ontology. A potential input to write such tests are the competency questions, i.e., the ontology requirements; however, how to write such tests from requirements and how to analyse the results are aspects that are out of scope of the method. The authors do not state whether the tests generated by the users are abstract and, therefore, reusable.

2.2.4. Fernández-Izquierdo and García-Castro's testing method

The same authors of this paper presented an ontology testing method based on requirements (Fernández-Izquierdo and García-Castro, 2018) that aims at systematising the generation and execution of tests cases extracted from ontology requirements. To that end, the authors proposed a testing process divided into three activities, i.e., *test design*, *test implementation* and *test execution*.

Ontology testing approaches are usually divided into two activities, i.e., *test implementation* and *test execution*. However, in this testing method the *test design* activity is needed due to the ambiguity and assumptions inherent to the natural language (Dennis et al., 2017) and to the fact that different people may be in charge of the design and implementation of tests. Therefore, in this design activity the goal of each requirement is identified and specified using a test language. Fig. 3 summarises these activities, together with their inputs and outputs.

It is worth mentioning that the tests defined during the test design activity and that are written using the language proposed in this method do not include information related to the ontology from which the tests are extracted. Consequently, these tests can be considered as abstract, since they are independent of any ontology.

Moreover, this testing method proposes the storage of the results of each activity as RDF files that follow the Verification Test Case vocabulary.[4] This ontology defines the relation between requirements and tests, as well as the relation between tests and their implementations and between the implementations and their execution on an ontology. Therefore, traceability between requirements, tests and ontologies can be obtained.

[3] The last version of TDDOnto is available in the following URL: https://github.com/kierendavies/tddonto2. The last update was on August 23, 2018.

[4] https://w3id.org/def/vtc#.

Table 1
Comparison between the conformance testing approaches.

Characteristic	ISO/IEC	ETSI	W3C
Activities definition	✓	✓	
Language definition	✓	✓	
Reusability support	✓	✓	
Good practices for specifying conformance clauses			✓
Conformance report	✓	✓	✓

This testing method is focused on the definition of a set of TPs based the ontology requirements, as well as on their implementation and execution on particular ontology. The tests proposed in this testing method are abstract, since they do not depend on the ontology on which they will be executed and they can be executed on multiple ontologies. However, the process of how to extract tests from a set of requirements in order to be representative according to what must be defined in the ontology and, therefore, to what must be satisfied by an ontology that should be conformant with it, is out of scope of this method. Moreover, the analysis of the tests results obtained after the execution of the tests on an ontology, which in a conformance scenario identifies the inconsistencies and overlaps between an ontology and a standard, is also out of scope of this method.

2.3. Comparison between conformance testing approaches

Conformance testing approaches developed by standardisation bodies agree on the fact that conformance testing is a key activity for ensuring the quality of products and for increasing the confidence in the correct functioning of an implementation with respect to a given specification.

Table 1 summarises the main characteristics of each approach, identifying their similarities and their differences. The symbol "✓" indicates that the characteristic is supported by the standardisation body.

Table 1 shows that while the ISO/IEC 9464 and the ETSI approaches are focused on describing how to write and reuse tests, e.g., developing a particular testing language, the W3C is focused on providing guidelines on how to specify conformance clauses.

As it was previously mentioned, in the Ontology Engineering field there is a lack of works on analysing conformance to check whether an ontology is compliant with the specification of a standard. However, an initial step may be the different ontology testing methods that allow to execute tests on ontologies. Table 1 summarises the main characteristics of the ontology testing approaches presented in this section. The symbol "✓" indicates that the characteristic is supported by the testing method.

Table 2 shows that although the majority of these testing approaches propose the definition of testing activities, these activities are oriented to the implementation and execution of tests in order to

Table 2
Comparison between the ontology testing approaches.

Characteristic	Blomqvist and colleagues	CQChecker	TDD	Fernández-Izquierdo and García-Castro
Activities definition	✓[a]		✓[a]	✓[a]
Test execution	✓	✓	✓	✓
Language definition				✓
Traceability between requirements and tests	✓			✓
Reusability support				✓
Conformance report				

[a]These activities are focused on test implementation and execution, but they do not support the entire workflow from requirements to results analysis.

verify whether an ontology satisfies certain conditions. The extraction of features to be tested and the associated requirements, the definition of abstract tests from the requirements, as well as the generation of the report with the result of the testing process, are out of scope of these testing methods. As it can be observed in the conformance testing approaches in the state of the art, these activities are crucial in an ontology conformance scenario in order to generate the tests that allow to build a conformance report. Such conformance report must show the results of the conformance testing process, identifying the inconsistencies and overlaps between the ontology and the standard.

The work presented in this paper is inspired by the conformance testing methods in the state of the art for Software Engineering and industry and, taking as input existing ontology testing methods, aims at providing a first step in analysing conformance in the Ontology Engineering field. Similarly to the ISO/IEC 9464 and the ETSI approaches, the ontology conformance testing and the minimum common identification methods proposed in this work are focused on the language definition for generating homogeneous tests and on their reusability.

3. Method for ontology conformance testing

Borrowing the concept from the Software Engineering field (Moseley et al., 2003), the conformance testing of ontologies refers to the purpose of determining up to what extent a particular ontology conforms to the specification of a standard. In this context, the standards to be conformed to could be standard ontologies as well as data models defined in standards supported by standardisation bodies, such as the ISO/IEC norms.

Following the conformance concepts described in the already introduced ISO/IEC 9646, any ontology conformance testing approach should include a testing process for designing, implementing and executing tests associated with a standard. Such tests, which are extracted from the standard specification to check a set of selected features, should be abstract in the sense that they do not include any information about the ontology in which the test will be executed, being independent of any ontology and, consequently, reusable. In the context of Ontology Engineering, a separated activity for analysing the results should be included, in order to determine up to what extent an analysed ontology is conformant to a particular standard and to generate a conformance report with the results.

This paper advances the state of the art in conformance testing for ontologies by proposing a conformance testing for OWL ontologies method based on functional requirements.[5] Functional ontology requirements, usually written in the form of competency questions (Grüninger and Fox, 1995) or statements, refer to the particular knowledge to be represented by an ontology (Suárez-Figueroa et al., 2009). Ontology developers that use this method must have knowledge about the OWL language (Hitzler et al., 2009), which is based on Description Logics (Baader et al., 2003), in order to identify relationships between concepts such as value restrictions, existential quantifications, or concepts inclusions and equivalences.

This ontology conformance testing method based on ontology requirements includes five activities, namely, *functional ontology requirements extraction*, *test design*, *test implementation*, *test execution* and *test results analysis*. Fig. 4 illustrates this method, together with the inputs and outputs of the different activities. This method starts with the extraction of functional ontology requirements from a standard, which are used to define the tests purposes. A test catalogue that lists all the supported tests is provided to help developers during the definition of test purposes. Once the developers have the test purposes defined, they have to implement them to be executed on a particular ontology. A glossary of terms of the ontology which maps each term in the test with each term in the ontology is also needed to execute such tests. Finally, the results obtained in the execution activity are analysed in the test analysis activity.

Since ontology verification methods already cover the design, implementation and execution of tests, this work builds on top of them. Hence, the ontology conformance testing approach proposed in this work is grounded on the existing testing method for ontology verification (Fernández-Izquierdo and García-Castro, 2018) proposed by the same authors of this paper, which was described in Section 2.2. However, the focus of these two works are different: while in the ontology verification method the main goal is to verify whether an ontology satisfies the expected requirements, in the conformance testing method the main goal is to check up to what extent an ontology satisfies the specification of a standard, allowing the identification of overlaps and differences between them. Furthermore, when a standard is revised or errata are issued, tests must be updated accordingly and the conformance testing revised. Therefore, although the conformance testing method is based on the ontology testing method, it was adapted and extended to support the workflow required in a conformance scenario.

3.1. Activities within the conformance testing method

The following sub-sections detail the testing activities that should be carried out in the proposed conformance testing framework. The definition of the test design, test implementation and test execution activities are reused from the testing method for ontology verification proposed by the same authors of this work, which provides an automatic procedure to implement and execute tests. The activities related to the extraction of requirements, the analysis of the results and the conformance report are defined in this paper for supporting the conformance analysis.

The conformance testing method is supported by the online tool Themis (Fernández-Izquierdo and García-Castro, 2019), which allows test implementation and test execution on one or multiple ontologies. However, the requirements extraction, the test design and the analysis of the results should be done manually.

3.1.1. Functional ontology requirements extraction

Functional ontology requirements extraction refers to the activity of collecting the functional ontology requirements that the standard specifies. These requirements should be related to the domain that the standard should model. To analyse conformance, they should define the concepts, relations and restrictions that must be defined in the domain described by the standard.

This activity produces a set of functional ontology requirements that could be materialised in one of the following forms:

- **Competency questions.** This technique was proposed by Grüninger and Fox (1995) and suggests establishing a set of queries that an ontology must be able to answer, e.g. "*What is an IoT gateway?*".

[5] Through this paper functional ontology requirements are referred to as ontology requirements for clarity.

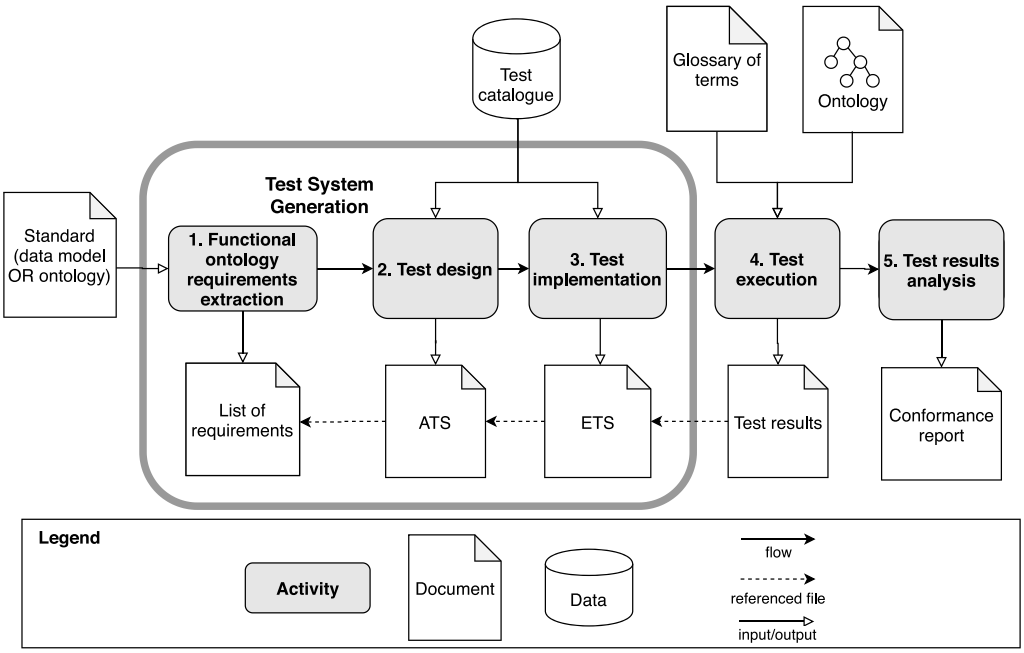


Fig. 4. Ontology conformance testing approach in Ontology Engineering.

- **Natural language statements.** In this case, the requirements are described by means of natural language statements, e.g., "*An IoT gateway is a digital entity*".

To ease the requirements proposal, the CORAL Corpus (Fernández-Izquierdo et al., 2019) can be used. CORAL includes a dictionary of lexico-syntactic patterns that identifies different types of ontology requirements and how they are specified, e.g., a requirement with the structure *What is NP⟨class⟩?*, asks about the existence of a class in the ontology named *NP⟨class⟩*. Moreover, CORAL also identifies a set of ambiguous expressions that can lead to multiple implementations in the ontology and, therefore, they should be avoided in the specification of requirements. As an example, the use of the verb "to have" in a requirement can be translated both as a datatype property and an object property in the ontology. In addition to these lexico-syntactic patterns, CORAL collects 834 requirements extracted from real-world ontologies that are annotated according to their lexico-syntactic patterns. Such requirements can be used as examples of how to write requirements.

Ontology requirements should be grouped in one or more topics related to them, e.g., the requirement "*What is a gateway?*" can be associated with the topic "*Gateway*". This grouping process facilitates the analysis of conformance once the test results are obtained, and allows to check conformance with regards to topics related to the ontology. To that end, each requirement should be tagged according to its associated topic.

Ontology developers must guarantee that the defined requirements are correct and complete since they will be used to check conformance with the associated ontology. However, the CORAL corpus shows that multiple requirements that are published online have ambiguous expressions that can lead to several implementations in an ontology. In that situation, the results obtained by the conformance analysis could be incorrect. Therefore, the following criteria defined by Grüninger and Fox (1995) can be used to check the status of the set of ontology requirements. These criteria do not aim to be an exhaustive list:

- A set of requirements is correct if each requirement refers to some features of the ontology to be developed.
- A set of requirements is internally consistent if no conflicts exist between them.
- A set of requirements is concise if each and every requirement is relevant, and no duplicated or irrelevant requirements exist.
- A set of requirements is realistic if each and every requirement meaning makes sense in the domain.
- Each requirement in the set of requirements is understandable to end-users and domain experts.
- Each requirement in the set of requirements is unambiguous if it has only one meaning; that is, if it does not admit any doubt or misunderstanding.

In a conformance testing scenario, the higher the quantity of requirements and the more concise and unambiguous the requirements are, the more accurate the identification of overlaps and conflicts between the standard and the ontology to be analysed.

3.1.2. Test design

During the test design activity, a set of test designs are created for each of the topics defined in the previous step. To that end, the desired behaviour of each standard requirement, i.e., their goal in the standard, is extracted and formalised into a set of supported test purposes (TPs), which describe the objective of testing. These TPs are defined without any information related to the ontology in which such TP will be executed, e.g., URIs or labels. Therefore, they can be considered as abstract TPs that allow their execution on multiple ontologies, including those that are not related to the standard from which they are extracted. Each test design includes the identifier, the TP, the requirement and the associated topic.

The set of test designs constitutes the Abstract Test Suite (ATS), which is the output of this test design activity. The ATS must be stored in an RDF file by using the Verification Test Case ontology,[6] which describes test suites and test designs, including all the information related to them, such as the requirement associated, the TPs, the tag according to the requirement topic and the standard associated with the test. This ATS should be published online so that developers can use it to check whether their ontologies pass such tests and, thus, are conformant to the standard. Moreover, published ATSs can also be understood as formal documentation of the standard.

To generate such ATS, each TP uses keywords that indicate its goal. These keywords are inspired by the software engineering approach

[6] https://w3id.org/def/vtc#.

called keyword-based testing (Roth, 2013). The idea behind keyword-based testing is to express each test as abstractly as possible, but precise enough to be executed and interpreted by a test execution tool. Therefore, based on the dictionary patterns identified by the CORAL Corpus, a set of keywords and TPs are identified to formalise the requirements provided the standards.

The complete set of TPs is presented in Table 3, which shows the identified goal of each TP and its corresponding syntax. In Table 3, those terms that are represented between brackets (e.g., "[Class]") correspond to those terms that should be completed by the user, while those terms italicised (e.g., *type*) correspond to the keywords that cannot be changed in the test expression. This set of TPs extends the one presented in the testing method for ontology verification (Fernández-Izquierdo and García-Castro, 2018) by adding new keywords. After analysing several standard specifications, e.g. the ISO/IEC 30141 (ISO/IEC 30141:2017, 2017) or the W3C SSN (Haller et al., 2019), it was found that current TPs (Fernández-Izquierdo and García-Castro, 2018) are not enough to formalise the requirements defined by them.

As an example, the requirement included in the ISO/IEC 30141 specification that states "*An IoT gateway is a digital entity*" is associated with the TP "*Gateway subClassOf DigitalEntity*". This TP has the goal of representing a subsumption relation between two classes, i.e., Gateway and DigitalEntity, as shown in Table 3.

3.1.3. Test implementation

The ATS defined during the test design activity must be implemented into queries or axioms to be executed on an ontology and to check whether it satisfies the standard requirement. The set of test implementations associated with the ATS constitutes the Executable Test Suite (ETS). The tests in the ETS are still abstract since they do not include any information about the ontology on which the tests are going to be executed; the same implementation can be executed on multiple ontologies.

The separation between the test design and the test implementation allows increasing the maintainability of tests. It is possible to change the implementation of tests without changing the test designs and, therefore, without changing the ATSs associated with standards. The ETS can also be stored in an RDF file by using the Verification Test Case ontology, which describes all the information related to how a test design is implemented.

The structure of the test implementation of each test design in the ATS is reused from the verification testing method (Fernández-Izquierdo and García-Castro, 2018). Each test implementation must include:

- A **precondition**, which is a SPARQL query that checks whether the terms involved in the test design are defined in the ontology to be analysed. This precondition allows to check whether the terms defined in the test design associated with the standard are defined in the ontology. If these terms are not defined in the ontology, then the test design is not satisfied.
- A set of **axioms to declare auxiliary terms**, which are a set of temporary axioms added to the ontology to declare the auxiliary terms needed for executing the TP included in the test design.
- A set of **assertions to check whether the ontology behaves as determined in the test design**, which are a set of pairs of axioms and expected results that represent different ontology scenarios. For each pair, the axiom is temporary added to the ontology to force a scenario, after which a reasoner is executed. The expected result determines if the ontology status (i.e., inconsistent ontology, unsatisfiable class or consistent ontology) after the addition is the expected one in case the test design was satisfied. If the ontology status concurs with the expected status, then the test design is satisfied. This assertion allows to check whether the knowledge expected by the standard requirement is included in the ontology.

Table 4 shows an example of test implementation for the standard requirement previously mentioned "*An IoT gateway is a digital entity*", which is associated with the TP "*Gateway subClassOf DigitalEntity*". The assertions are written using Description Logics syntax (Baader et al., 2004).

3.1.4. Test execution

Once the ETS is generated, each test implementation can be executed on an ontology. This execution activity consists of three steps: (1) the execution of the query that represents the preconditions, (2) the addition of the axioms which declare the auxiliary terms, and (3) the addition of the assertions. After the addition of each axiom, a reasoner is executed to report the status of the ontology, i.e., whether the ontology is consistent, inconsistent or has unsatisfiable classes. The addition of the auxiliary axioms needs to always lead to a consistent ontology. However, in the case of the assertions, the agreement between the reasoner status after the addition of all the axioms and the status indicated in the test implementation determines whether the ontology satisfies the TP.

During this activity, each test implementation in the ETS should be first completed with the information related to the ontology to be executed. To that end, a glossary of terms must be generated manually or automatically to map each term in the standard test with a term in the ontology. Therefore, the terms in the TP that are defined in the ontology in which the test implementation will be executed, e.g., *Gateway*, are collected and associated with a URI in the ontology, e.g., *http://example.org/ontology#Gateway*. Then, using these associations, the terms in the test implementation are translated into a term in the ontology. Consequently, the ETS can be executed on the ontology although it was extracted from a standard specification document.

Following the ontology verification testing method (Fernández-Izquierdo and García-Castro, 2018), the conformance testing method provides four possible results for each test implementation in the ETS and each ontology:

1. *Passed*: if the ontology passes the preconditions defined in the test implementation and the results of the assertions are the expected ones. This result indicates that the ontology satisfies the test design associated with the standard.
2. *Undefined terms*: if the ontology does not pass the preconditions. This result indicates that the terms included in the test design associated with the standard are not defined in the ontology.
3. *Absent relation*: if the ontology passes the preconditions and the results of the assertion are not the expected ones but there are no conflicts in the ontology. This result indicates that there is a relation defined in the test design associated with the standard that is not defined in the ontology.
4. *Conflict*: if the ontology passes the preconditions and the results of the assertion are not the expected ones, and the addition of the axioms related to the test design included in the associated test design leads to a conflict in the ontology. This result indicates that there is a conflict between what is defined in the standard and what is defined in the ontology.

As an example, the test implementation described in Table 4 can be executed on the ontology with URI *http://example.org/ontology#*. If the ontology defines the terms *Gateway* and *DigitalEntity*, as well as a subsumption relation between them, then the result of this execution should lead to the *passed* result and, therefore, the test design is passed and the associated standard requirement would be satisfied.

3.1.5. Test results analysis

To extract more information from the tests and to determine up to what extent an ontology conforms to a standard, a new activity is defined in this paper, i.e., *test results analysis*. During this activity, the results of the test execution are analysed to identify the degree of conformance of the ontology regarding the standard.

Table 3
List of the test purposes supported by the conformance testing method.

Goal of the test	Syntax of the test
T1. The ontology contains the class A	[ClassA] *type Class*
T2. Subsumption relation between classes A and B	[ClassA] *subClassOf* [ClassB]
T3. Disjointness between two classes	[ClassA] *disjointWith* [ClassB]
T4. Equivalence between two classes	[ClassA] *equivalentTo* [ClassB]
T5. The ontology contains the property P	[Property] *type Property*
T6. Existential relation P between two classes A and B	[ClassA] *subClassOf* [PropertyP] some [ClassB]
T7. Universal relation P between two classes A and B	[ClassA] [PropertyP] *only* [ClassB]
T8. Symmetric property P	[PropertyA] *characteristic symmetricProperty*
T9. Minimum cardinality.	[ClassA] *subClassOf* [PropertyP] *min* [num] [ClassB]
T10. Maximum cardinality.	[ClassA] *subClassOf* [PropertyP] *max* [num] [ClassB]
T11. Cardinality.	[ClassA] *subClassOf* [PropertyP] *exactly* [num] [ClassB]
T12. Universal relation P between the union of two classes A and B	[ClassA] [PropertyP] *only* [ClassB] *or* [ClassC]
T13. Universal relation P between the intersection of two classes A and B	[ClassA] [PropertyP] *some* [ClassB] *and* [ClassC]
T14. The ontology contains the individual I	[IndividualI] *type* [ClassA]
T15. Multiple inheritance of class A	[ClassA] *subClassOf* [ClassB] *and* [ClassC]
T16. Subsumption and relation between classes	[ClassA] *subClassOf* [ClassB] *that* [PropertyP] *some* [ClassC]
T17. Minimum cardinality between classes A and B, and existential relation P between classes B and C	[ClassA] *subClassOf* [PropertyP] min [num] [ClassB] and [ClassB] *subClassOf* [PropertyP] *some* [ClassC]
T18. Minimum cardinality between classes A and B and universal relation P between classes B and C	[ClassA] *subClassOf* [PropertyP] *min* [num] [ClassB] and [ClassB] [PropertyP] *only* [ClassC]
T19. Subsumption relation between A and B, subsumption relation between A and C, and disjointness between B and C	[ClassA] *subClassOf* [ClassB] *and* [ClassC] *subClassOf* [ClassB] *that disjointWith* [ClassA]
T20. Participation ODP between classes A and B	[ClassA] *subClassOf* [participates] *some* [ClassB]
T21. CoParticipation ODP	[ClassA] and [ClassB] *subClassOf* [participates] *some* [ClassC]
T22. Part-whole relationship (existential restriction)	[ClassA] *subClassOf* [isPartof] *some* [ClassB]
T23. PartOf ODP between classes A and B (universal restriction)	[ClassA] *subClassOf* [isPartof] *only* [ClassB]
T22. Participation ODP between classes A and B	[ClassA] *subClassOf isParticipantIn—hasParticipant some* [ClassB]
T23. CoParticipation ODP	[ClassA] *and* [ClassB] *subClassOf isParticipantIn—hasParticipant some* [ClassC]
T24.PartOf ODP between classes A and B. Existential restriction	[ClassA] *subClassOf isPartof—hasPart some* [ClassB]
T25. PartOf ODP between classes A and B. Universal restriction	[ClassA] *subClassOf isPartof—hasPart only* [ClassB]
T26. Object-Role ODP between classes A and B. Existential restriction	[ClassA] *subClassOf isRoleOf—hasRole some* [ClassB]
T27. Object-Role ODP between classes A and B. Universal restriction	[ClassA] *subClassOf isRoleOf—hasRole only* [ClassB]

Table 4
Test implementation of a test for checking subsumption between the classes Gateway and DigitalEntity.

Requirement:	An IoT is a digital entity	
Test purpose:	Gateway subClassOf DigitalEntity	
Type:	T2. Subsumption between classes	Related to: Classes
Test precondition		Test preparation
Class Gateway and Class DigitalEntity exist		(S 1.1) Declaration of ¬Gateway (S 1.2) Declaration of ¬DigitalEntity
Assertions to test the ontology behaviour		
Axiom		Expected status after adding the axiom
(S 2) Gateway' ⊑ ¬Gateway ⊓ DigitalEntity		Consistent ontology
(S 3) Gateway' ⊑ Gateway ⊓ ¬DigitalEntity		Unsatisfiable class
(S 4) Gateway' ⊑ Gateway ⊓ DigitalEntity		Consistent ontology

To analyse such conformance, the following information extracted from different artefacts involved in the verification process, i.e., the requirements, the tests and the ontology, can be obtained:

- How many tests included in the ATS associated with the standard are satisfied by the ontology. The *Passed Test Percentage* metric should be calculated to obtain this information. The higher the percentage of this metric, the greater the coverage of the ontology regarding the standard specification document.
- How many terms defined in the ATS associated with the standard are out of scope of the ontology. The *Undefined Test Percentage* metric should be calculated to obtain this information. The higher the percentage of this metric, the greater the number of standard terms that are out of scope of the ontology.
- How many relations in the ATS associated with the standard are not included in the ontology. The *Absent Test Percentage* metric should be calculated to obtain this information. The higher the percentage of this metric, the greater the absent relations and restrictions in the ontology.
- How many tests in the ATS associated with the standard are not passed by the ontology. The *Failed Test Percentage* metric should be calculated to obtain this information. The higher the percentage of this metric, the greater the conflicts between the standard and the ontology.
- How many requirements in a standard specification document are satisfied by the ontology, since sometimes a requirement needs more than one test to be checked. The *Covered Requirements Percentage* metric should be calculated to obtain this information. The higher the Covered Requirements Percentage metric, the greater the coverage between the standard and the ontology.
- How many requirements associated with a standard specification have a conflict with the ontology. The *Requirements Fault Percentage* metric should be calculated to obtain this information, which in this case refer to those requirements of the standard that have a conflict with the ontology. Similarly to the Test Failed Percentage metric, the higher the percentage of this metric, the greater the conflicts between the standard and the ontology.

- How many terms included in the standard are also defined in the ATS. In the conformance testing scenario, having this information helps analysing the completeness of the requirements specification used in the testing process. The *Number of Tested Vocabulary Terms* metric should be calculated in order to obtain this information. The higher the percentage of this metric, the more complete the requirements specification will be. However, it should be considered that due to modelling decisions or ontology design patterns (Gangemi and Presutti, 2009) new terms can be added to the ontology but avoided in the requirements.

Using the previous metrics, the conformance degree can be determined through the following steps. Each step can be related to the requirements topics (topic level) or to the entire set of requirements (general level):

1. To identify the requirements of the standard whose tests are passed by the ontology, i.e., the coverage between the standard and the ontology. The Passed Test Percentage and Covered Requirements Percentage metrics facilitate the identification of such coverage. The Number of Tested Vocabulary Terms metric provides an overview of completeness of the requirements specification, which can also be included in the coverage analysis.
2. To identify the requirements whose tests result in *undefined terms*, i.e., requirements that include information that is included in the standard but is out of the scope of the ontology to be analysed.
3. To identify the requirements whose tests result in *absence*, i.e., requirements that include information related to restrictions and relations that are defined in the standard but not in the ontology. The Absent Test Percentage metric provides an overview of the percentage of tests with absent relations.
4. To identify the requirements whose tests result in *conflict*, i.e., the incompatibilities between the standard and the ontology. The Failed Test Percentage and Requirement Fault Percentage metrics identify the percentage of incompatible tests and requirements between the standard and the ontology to be analysed.

As an example of the degree of conformance that can be obtained with this information, if a set of requirements for a standard is divided into three topics, e.g., (1) Devices, (2) Users and (3) Services, it could be determined that the ontology to be analysed has a Covered Requirements Percentage of 60% for the first topic, but a 0% for the second and third topic. In this case and with this information it can be concluded that there is coverage between the standard and the ontology only for the specification of devices.

Based on these results, the conformance report is generated. The goal of this conformance report is to inform about the degree of conformance of a given ontology by using all the information obtained during the conformance analysis. Inspired by the ISO/IEC 9646, such report includes the following fields:

- The ontology to be analysed.
- The standard for which the analysis has been performed.
- The ATS together with its results.
- The ETS generated from the ATS.
- The glossary of terms used during the execution of the ETS.
- The obtained metrics for each topic or for the complete list of requirements.
- A clause that states the conformance status, which could include the overall conformance and the conformance divided by topic.
- A clause that provides information about those requirements related to the standard that are in conflict with the ontology.

The information provided by the conformance report can be used for requesting changes or for identifying conflicts between the developed ontology and the standard. Moreover, it can also be used to identify mappings and potential terms for reuse, as well as for guaranteeing interoperability between the standard, or standard topics, and the ontology.

4. Minimum common knowledge identification method

When analysing a particular domain, it can be observed that several standards coexist. As an example, in the IoT field the ETSI developed the SAREF ontology for smart applications (ETSI, 2017), the W3C developed the SSN ontology for describing sensors, their observations, and related procedures (Haller et al., 2019), and the oneM2M organisation developed the oneM2M ontology for providing syntactic and semantic interoperability of oneM2M platforms with external systems (oneM2M, 2016). All these mentioned ontologies are related to the IoT domain; however, they are oriented to the description of different aspects in such domain.

Since reusing other ontologies is a key activity during ontology development, ontology developers should be aware of how a set of standards covers their domain, identifying their scope, their overlaps, their differences and their conflicts. This information is valuable for identifying potential mappings and needs that are not supported by existing standards or that are defined in more than one. To that end, this section introduces a method for common knowledge identification between a set of standards which identifies the overlaps that are shared among their specifications. These standards can be ontologies or non-ontological specification documents that define data models.

This minimum common knowledge identification method is grounded on the conformance testing method described in the previous section. The activities related to the test design, implementation and execution are reused from the testing method for ontology verification. However, the test execution activity is adapted to the particular scenario of minimum common knowledge identification to support the execution of all the ATSs on all the standards. The generation of ontologies for data models without an ontology associated, the generation of shared glossaries of terms and the analysis of the results are novel activities defined in this paper. A more detailed description related to the differences between these two methods is described in the following sub-sections.

4.1. Activities within the minimum common knowledge identification method

To carry out this minimum common knowledge identification method, the functional ontology requirements associated with all the standards to be analysed should be extracted to generate an ATS from each of them. Then, these ATSs are executed on all the standards, and the obtained results are analysed to determine the overlaps and differences between them.

The overview of the method for identifying the minimum common knowledge between standards is summarised in Fig. 5. This method is grounded on the conformance testing method presented in Section 3 and, consequently, the activities are similar. However, in this scenario, all the ATSs are executed on all the standards.

4.1.1. Generation of ontologies from data models

The main challenge of this method is to execute the ATSs on every standard, even on those without any associated ontology, e.g., data models described in ISO/IEC norms. Therefore, an ontology is built from the ATS related to the standards that are not specified through an ontology to solve this issue.

The goals of the supported TPs that can be included in an ATS (Table 3) and that indicate the knowledge that should be defined in the standard, such as a subsumption relation between two classes, were

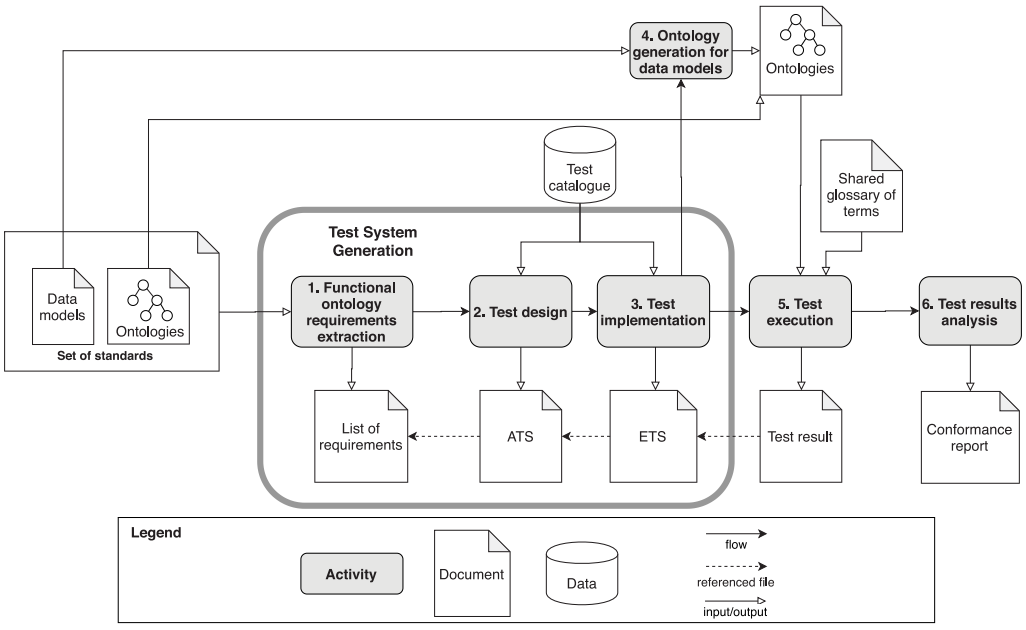


Fig. 5. Conformance approach for identifying the minimum commitment between ontologies.

collected and analysed to be able to automatically build an ontology from an ATS. Based on this expected knowledge, a set of OWL axioms was associated with each supported TP. Table 5 shows an example of an association between TPs and axioms for three different TPs.

These associations between a TP and a set of OWL axioms allows to automatically generate an ontology from an ATS for those standards that do not have an ontology related. The collection of all the OWL axioms retrieved by all the TPs in an ATS constitutes the ontology to be used in the method.

To support part of the method, an extension of Themis[7] has been developed that, before executing the tests, automatically generates ontologies from the associated ATS for those standards without a related ontology. This extension of Themis also generates automatically a shared glossary of terms that should be used for the execution of the ATSs. The requirements extraction, the test design and the analysis of the results are out of scope of this extension of Themis and should be performed manually.

4.1.2. Generation of a shared glossary of terms

A shared glossary of terms maps the terms in each TP included in the ATSs and the terms in each standard. This glossary of terms is required to automatically execute all ATSs on all the standards since it provides the required information to complete the abstract tests with the corresponding URIs for each standard. Table 6 depicts an example of a shared glossary of terms.

This glossary of terms should be generated either automatically or manually by the person in charge of the conformance testing process, who has to ensure that the glossary of terms is correct because an incorrect glossary of terms can lead to mistakes in the method results. It should be noted that the glossary of terms is a crucial element in the minimum common knowledge identification method since it allows to map each term in the ATSs with the terms in the standard.

4.1.3. Analysis of the results

The output of the execution activity consists of the results obtained after the execution of each ATS in each standard. The amount of results hinders the test result analysis since more results have to be analysed. However, the same steps that were detailed in Section 3.1.5 can be followed for analysing those results. The adoption of this method allows the developers to identify different layers of knowledge between standards, namely (Cuenca et al., 2019):

- The *common knowledge*, which represents the knowledge shared by the set of analysed models. In this case, the common knowledge represents the requirements that are satisfied by all the ontologies to be analysed.
- The *variant-domain knowledge*, which represents the knowledge that is common to more than one model, even though it is not shared by all of them. In this case, the variant-domain knowledge represents the requirements that are satisfied by more than two ontologies of the set of ontologies to be analysed.
- The *domain-task knowledge*, which represents the lower and most specific knowledge. In this case, the domain-task knowledge represents the requirements that are satisfied by only one ontology of the set of ontologies to be analysed.

These layers determine the knowledge that is common and the knowledge that is particular for each standard, identifying the scope and the commonalities between a set of standards. In addition to these layers, during the analysis of results the requirements that create conflicts between standards can also be identified.

5. Validation

To provide an assessment of the conformance and the minimum common identification methods and to address the research questions presented in Section 1, an empirical analysis has been carried out in a concrete use case. From the research questions, two hypotheses have been proposed that aim to be validated with this analysis. Considering the numerous standards that are proposed for the same domain, and that each standard describes a particular area of such domain, the hypotheses to be validated state:

H1. The common and variant-domain knowledge between ontologies and standards related to the same domain represent a small percentage of the requirements in the requirement specification documents of the standards.

[7] The code and distribution of the extension are available in the GitHub repository: https://github.com/albaizq/ThemisForConformance.

Table 5
Example of association between tests and axioms.

Goal	TP	Axioms
T1. Class definition	[ClassA] type Class	:A rdf:type owl:Class .
T2. Subsumption relation between classes A and B	[ClassA] subclassOf [ClassB]	A rdf:type owl:Class :B rdf:type owl:Class; rdfs:subClassOf :A .
T3. Disjointness between two classes	[ClassA] disjointWith [ClassB]	A rdf:type owl:Class :B rdf:type owl:Class; owl:disjointWith :A .

Table 6
Example of a shared glossary of terms.

Term in test	SAREF ontology	SSN ontology
Actuator	https://saref.etsi.org/core/Actuator	http://www.w3.org/ns/sosa/Actuator
Sensor	https://saref.etsi.org/core/Sensor	http://www.w3.org/ns/sosa/Sensor

H2. The common and variant-domain knowledge between ontologies and standards related to the same domain are related to general requirements, such as definition of concepts, while more restrictive requirements, such as cardinalities or functional properties, represent domain-task knowledge.

To validate the hypotheses, two experiments were performed: (1) following the conformance testing method described in Section 3, it was analysed the conformance between an ontology and a set of standards related to the same domain; and (2) following the minimum common knowledge identification method proposed in Section 4, the minimum common knowledge between the aforementioned standards was analysed. The first experiment provided insights about the hypotheses, while the latter one confirmed them.

5.1. Conformance testing

The goal of this experiment was to determine which type of requirements related to standards were satisfied by a given ontology. To that end, an ontology network already published on the Web was collected, together with a set of standards related to the same domain.

These standards needed to have associated requirements that identify the knowledge defined in them or documentation that allows extracting the requirements. However, the standards did not need to have an ontology associated. Afterwards, the tests related to these requirements were defined following the conformance testing method described in Section 3. Moreover, as explained in Section 3, these tests were abstract and did not have any information about ontologies, such as URIs, making them independent of the ontology from which they are extracted. Therefore, these tests can be executed on any ontology even though they were extracted from standards.

To analyse the type of shared requirements, the conformance testing method described in Section 3 was applied. In this method, tests associated with the requirements of each standard are executed on an ontology. Those requirements that are satisfied by the ontology are considered as shared requirements.

The following steps summarise the actions performed during the experiment:

1. To gather the functional requirements related to the standards.
2. To group them according to the topic associated, as described in Section 3.1.1.
3. To generate the ATS associated with these requirements following the testing process and testing language described in Section 3.1.2.
4. To group the tests designs in the ATS according to the type of TP and topic associated.
5. To execute the ATS on the ontology network.
6. To analyse the results in order to identify classes and properties that are shared between each standard and the ontology.

The following information was retrieved from the performance of these steps for each ontology in the ontology network:

- The types of the tests associated with the requirements of the standards that are passed by the ontology.
- The topics related to the tests associated with the requirements of the standards that are passed by the ontology.
- The results of the tests associated with the requirements of the standards that were executed on the ontology.
- The classes and properties that are shared between each standard and the ontology to be analysed based on the results of the tests.

To support the execution of tests, the Themis tool was used. Themis executed all the tests associated with the standards on each ontology and identified which were the tests that were passed by the ontology and which were not. Afterwards, a manual analysis to identify the shared types, topics and terms was performed.

5.1.1. Results

The ontology network that was developed in the European VICINITY project was selected for this experiment.[8] This ontology network aims to provide interoperability in the IoT domain and includes five ontologies, i.e., the VICINITY Core (Core), the Web of Things (WoT), the WoT mappings (Mappings), the VICINITY Adapters (Adapters), and the Datatypes (Datatypes) ontologies. The Core ontology represents the information needed to exchange IoT descriptor data between peers through the VICINITY platform. The WoT ontology aims to model the Web of Things domain according to the W3C WoT Working Group's Things Description Model (Kaebisch et al., 2020). The Mappings ontology represents the mechanism for accessing the values provided by web things in the VICINITY platform. The Adapters ontology aims to model all the different types of devices and properties that can be defined in the VICINITY platform. Finally, the Datatypes ontology aims to model the required and provided datatypes that are used in the interaction patterns of the platform.

The VICINITY network is related to the IoT field. Therefore, the following IoT standards were selected for the experiment: (1) the ETSI SAREF ontology, which describes smart applications; (2) the W3C SSN ontology, which describes sensors, their observations, and related procedures; and (3) the oneM2M base ontology, which provides syntactic and semantic interoperability of oneM2M platforms with external systems. Moreover, the conformance with the following non-ontological standard data models in the IoT field was analysed: (1) the ISO/IEC 30141:2017 (ISO/IEC 30141:2017, 2017), which describes the IoT Reference Architecture for connecting systems; and (2) the OCF standard,[9] which describes devices and how they interact. It should be mentioned that neither the OCF standard nor the ISO/IEC 30141 have related ontologies.

[8] http://vicinity.iot.linkeddata.es/.
[9] https://openconnectivity.org.

Table 7
Summary of requirements information for the IoT standards.

Standard	Version	Requirements provenance	Requirements	Tests
ISO/IEC 30141	–	ISO/IEC 30141:Internet of Things (IoT) – Reference architecture (ISO/IEC 30141:2017, 2017)	35	35
OCF	v2.0.2	OCF core specification[a]	27	27
oneM2M	v3.6.0	SAREF extension investigation technical report (TR 103 411) (ETSI, 2017)	33	33
SAREF	v2.1.1	SAREF extension investigation technical report (TR 103 411) (ETSI, 2017)	68	68
SSN	V2.0	W3C SSN specification (Haller et al., 2017)	34	34

[a]https://openconnectivity.org.

Table 8
Summary of testing results of the IoT standards for the VICINITY ontology network.

Standard	Test results			
	Passed	Undefined term	Absent	Conflict
ISO/IEC 30141	12 (34.26%)	19 (54.28%)	4 (11.42%)	0
OCF	9 (33.33%)	17 (62.93%)	1 (3.70%)	0
oneM2M	2 (6.06%)	30 (90.90%)	1(3.03%)	0
SAREF	7 (10.29%)	61 (89.70%)	0	0
SSN	12 (35.29%)	11 (32.35%)	11 (32.35%)	0
Total	**42 (21.32%)**	**138 (70.00%)**	**17 (8.63%)**	**0**

The requirements related to these standards were collected in the form of both competency questions and natural language statements, and the associated tests were defined by using the test catalogue depicted in Table 3. Table 7 shows the list of standards, the provenance of the gathered requirements associated with such standards, and the number of ontological requirements and defined tests per standard. As it is shown, some of the requirements were extracted from official documentation, e.g., Technical Reports, while others were extracted from online specifications. The list of gathered requirements, as well as the test suites generated from them, are available in the VICINITY ontology portal.[10] In this use case, it was defined as one test per requirement. In total, 197 tests were specified.

The W3C SSN, the OCF and the ISO/IEC 30131 do not have requirements defined in their official documentation and specifications. Therefore, based on the documentation associated with the standards and following the process defined in Section 3.1.1, a set of requirements were extracted for each one. These requirements identified the classes, properties and relations between classes that are defined in each standard. For the W3C SSN standard, the requirements are mainly related to observation and actuation, therefore, not all the classes and properties defined in the standard are included in the list of requirements.

Themis was used to check whether the VICINITY ontology network satisfies the 197 tests defined for the IoT standards, which would indicate the conformance of the ontology network with regards to those standards. Since the tests of such standards were generated following the testing method proposed in Section 3, the tests are abstract and do not have any information of ontologies such as URIs, making them independent of the ontology from which they are extracted. Therefore, these tests can be executed on the five ontologies in the VICINITY ontology network. Table 8 shows the testing results for each standard after the Themis execution.

From the results obtained by following this ontology conformance testing method, it can be deduced that the VICINITY ontology network passes 42 tests related to the standards, but does not take into consideration some concepts related to them as it is shown in the 138 requirements with the *undefined term* result, which determine those requirements that include concepts that are not defined in the ontology. However, it can also be concluded that there are no conflicts between the VICINITY ontology network and these standards since there are no tests with the *conflict* result, even though there are some absences, i.e., the terms specified in the standard are defined in the VICINITY ontology network but the relations between the terms are not. Therefore, it can be concluded that there are no inconsistencies between the domain defined in the standards and the domain defined in the VICINITY ontology network.

Table 8 also shows in parentheses the Passed Test Percentage, Undefined Test Percentage and Absent Test Percentage metrics for each standard, from which it can be observed that the majority of the tests result in *undefined term*. Those undefined terms in the tests identify the terms that are out scope of the VICINITY domain. As an example, the SAREF ontology has 89.7% of Undefined Test Percentage, which reflects that 89.7% of the requirements defined in the SAREF specification have terms that are not included in the domain described by VICINITY. Therefore, from this table, it can be concluded that the overlap between the VICINITY ontology network and SAREF is minimum and that they describe different aspects of the IoT domain. The same happens with the oneM2M ontology, which has 90.9% of Undefined Test Percentage.

In the case of the ISO/IEC 30141, the OCF and the W3C SSN the Passed Test Percentage is higher than 30%, indicating that there are overlaps between them and the VICINITY ontology network. It should be considered that the VICINITY ontology network imports the SOSA ontology, which is a module included in the SSN ontology and, therefore, there is more overlap between them.

Table 9 was created to analyse the types of tests passed by the VICINITY ontology network. This table shows the Passed Test Percentage grouped according to the type of test, e.g., in the case of the SAREF ontology, the VICINITY ontology network passed 40% of tests related to the definition of classes. From this table it can be observed that the type of test with the highest Passed Test Percentage is the definition of classes, although there is a small number of this type of tests defined in the test suite. This occurs because in the requirements specifications there are only a small number of requirements related to the definition of classes. It can also be observed from Table 9 that the ontology network also satisfies requirements related to subsumption and relation between classes, although to a lesser extent. Furthermore, the tests related to cardinality constraints were not passed by the VICINITY ontology. This information reflects that the ontology network and the standards are more interoperable at the terms level rather than at the constraints level.

Additionally, Table 10 shows the topics related to the requirements [11] of the IoT standards and the percentage of passed tests by the VICINITY ontology network associated to the requirements of the topics of each standard. In this use case, each topic refers to a class of the ontology that can be extracted from each requirement together with its properties.

From Table 10 it can also be observed that the VICINITY ontology network passed at least one test of the majority of the SSN and ISO/IEC 30141 topics. However, in the case of the SAREF ontology, the VICINITY ontology network only passes 10.86% of the tests related to devices. For the rest of standards, it can be observed that the majority of topics has 0% of Passed Test Percentage, i.e., none of the tests related to those topics were passed by the VICINITY ontology. However, there are some topics with 100% of Passed Test Percentage, which indicates that the ontology network is compliant with the requirements from the standards associated to such topics.

[10] http://vicinity.iot.linkeddata.es/vicinity/conformance.html.

[11] Note that a requirement can belong to one or more topics.

Table 9
Summary of the types of tests passed by the VICINITY ontology.

Type of test	ISO 30141		OCF		oneM2M		SAREF		SSN	
	Total	Passed	Total	Passed	Total	Passed	Total	Passed	Total	Passed
Definition of classes	0	–	1	100%	6	16.67%	7	40%	8	87.5%
Subsumption between classes	7	85.71%	4	0%	4	0%	17	17.64%	3	66.67%
Relation between classes (universal and existential restrictions)	23	26.08%	21	38.09%	20	5%	40	5%	20	15%
Cardinalities	5	0%	1	0%	3	0%	6	0%	3	0%

Table 10
Summary of the topics of each standard and the Passed Test Percentage of the VICINITY ontology network.

Ontology	Topic	N° requirements	Passed test percentage
ISO/IEC 30141	Thing	9	60%
	Accessibility	9	21.43%
	Device	9	28.57%
	Service	8	25%
	User	4	100%
	Data store	2	0%
	Actuator	1	100%
OCF	Device	14	35.71%
	Resource	15	0%
	Endpoint	3	33.33%
	Thing	2	0%
	Aspect	1	0%
	Function	1	0%
	Link	1	0%
	Maintenance	1	0%
	Property	1	0%
	Security	1	0%
oneM2M	Service	7	0%
	Function	6	0%
	Operation	6	0%
	Device	5	100%
	Accessibility	4	0%
	Thing	3	33.33%
	Aspect	2	0%
	Communication	2	0%
	Task	2	0%
	Metadata	1	0%
	Network	1	0%
	Profile	1	0%
SAREF	Device	46	10.86%
	Function	19	0%
	Service	6	0%
	Property	5	0%
	Building	4	25%
	Command	3	0%
	Commodity	2	0%
	Profile	2	0%
	Price	1	0%
	Time	1	0%
SSN	Property	7	100%
	Stimulus	6	0%
	System	6	40%
	Feature of interest	5	40%
	Procedure	5	33.33%
	Sensor	5	75%
	Observation	4	20%
	Results	4	0%
	Deployment	3	50%
	Observable Property	3	50%
	Actuator	2	66.66%
	Sample	2	0%

Moreover, it can also be observed that the topics which had more passed tests are related to high-level terms, e.g., Device, Sensor, Actuator or Thing. These terms are used in the majority of scenarios in the IoT field, regardless of the area of concern to be described. However, those terms that are particular to a use case, e.g., Data store, Network or Metadata, were only included in those ontologies focused on that use case.

Finally, for illustrating the conformance between the VICINITY ontology network and the IoT standards, Fig. 6 was created, where arrows with white triangles on the top represent a subsumption relation between two classes. The origin of the arrow is the class to be declared as a subclass of the class at the destination of the arrow. Also, directed arrows are used to represent object properties between classes and parentheses represent cardinality restrictions. These figures show with dotted lines those properties and classes that are shared between each

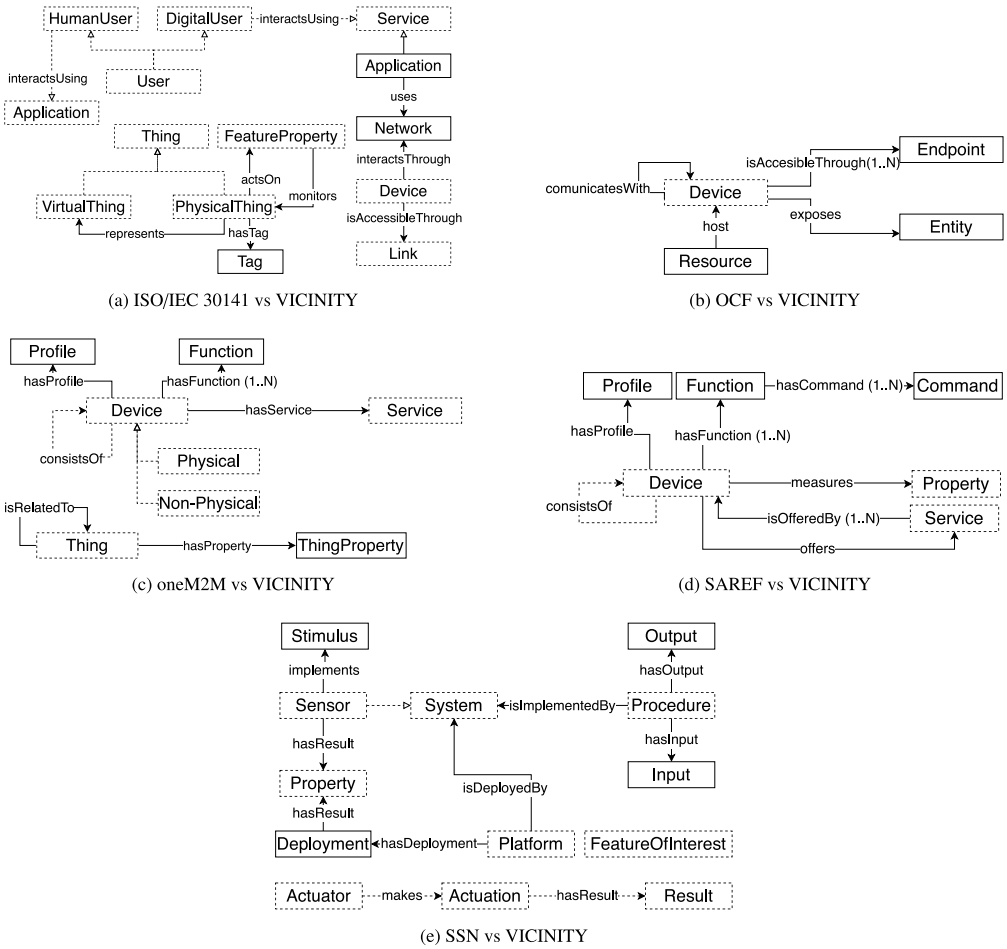


Fig. 6. Shared properties and classes between IoT standards and the VICINITY ontology network.

of the standard ontologies and the VICINITY ontology network. It is worth noting that the conceptualisations in Fig. 6 which represent the OCF and the ISO/IEC 30141 standards were generated from the functional requirements since there are no ontologies related to them.

After analysing all the results obtained by the conformance method, it can be concluded that there are several requirements shared between the analysed ontology and each standard. However, these shared requirements are mostly related to the definition of classes, as it is confirmed in Fig. 6. The more specific requirements, such as those related to cardinalities, were only satisfied by the standard from which the requirements were extracted, although there can be some exceptions for example if an ontology imports a standard. Hence, the common knowledge between the ontology and the standards represents only 21.32% of the standards' requirements.

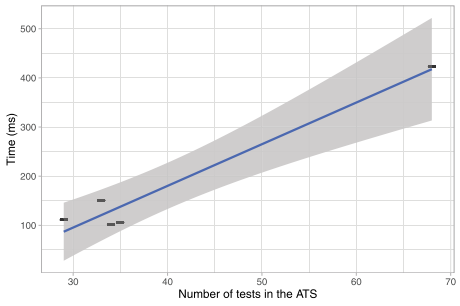
Not only those requirements related to the definition of tests are satisfied by the ontology, but also those related to relations between terms. However, the number of passed tests associated with these requirements is smaller. In this use case ontologies and standards share those simple requirements that are related to classes or relations. This situation can stem because more complex restrictions are defined for particular cases when describing a field, and, therefore, they are not common to any scenario of such field. Bear in mind that these results depend on the quality of the specification of requirements: the more requirements, the more accurate the conformance analysis.

To conclude, the results provided by Tables 8 and 9 indicate that the number of requirements shared by a set of standards and ontologies in the same domain is small. Moreover, these shared requirements are mostly related to the definition of classes, although requirements related to relations between terms are also shared. Table 8 shows that only 21.32% of the tests associated with the requirements of the IoT standards are satisfied by the VICINITY ontology. Moreover, from Table 9 it can be observed that the tests related to the definition of classes are the tests with the highest Passed Tests Percentage. With this, it could be observed that although a set of ontologies are related to the same domain they might not share the majority of their terms. Moreover, they might share the definition of some of their terms and relations, but not the particular restrictions associated with a particular use case. However, it is important to observe that, although they do not share all the requirements, they do not have conflicts between them.

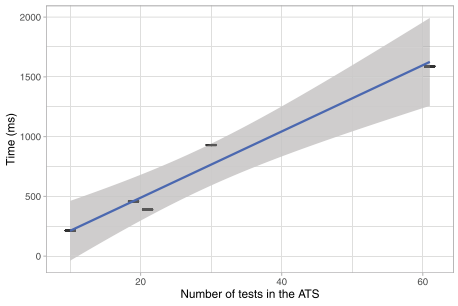
These results can be considered both by ontology developers and by users that use the VICINITY ontology network since they can check that such ontology network conforms to the definitions of the terms of well-known standards and that no conflicts were found, which increases its interoperability. Moreover, they can also be informed about which topics of the standards are covered by the ontology network and which are not. As an example, while the VICINITY ontology is conforming to the device specification provided by the oneM2M ontology, it does not consider any term related to security, although it is a topic included in the standard.

The developers of the VICINITY ontology could also benefit from the conformance analysis to define new requirements for the ontology, taking into account that the more interoperable ones might be those related to the definition of classes and properties rather than those requirements that are more restrictive. However, as shown in Table 9, requirements related to the definition of classes are not common in requirements specification documents from standards.

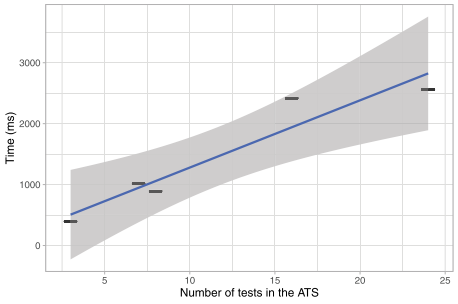
The results gathered from this use case indicate that hypotheses H1 and H2 could be validated. However, to have more information



(a) Loading time



(b) Execution time for tests with *undefined* results



(c) Execution time for tests with *passed, absent and conflict* results

Fig. 7. Time spent for the test suites related to the standards.

to validate them, the minimum common knowledge analysis has been performed.

5.1.2. Performance

To analyse the performance of the method, it was calculated the time spent loading and executing the tests using Themis. To only calculate the load and execution time, and not the time spent retrieving the data from the corresponding servers, both the tests and the VICINITY ontology were uploaded as files in Themis, rather than using the URIs. Fig. 7 depicts the total time spent loading and executing the test suite of each standard and how the number of tests influences such time.

The execution time of the ATS is divided into the execution time for those tests that result in *undefined terms* in the ontology and the execution time for those tests with full execution, i.e., those that result in *passed* by the ontology, *absent* or *conflict*. This division was made because those tests that result in *undefined terms* only execute the first of the three steps of the execution algorithm, i.e., the execution of the precondition, while the other ones execute the three steps.

To check the scalability of the method, a linear regression model (Muggeo, 2008) was applied with a confidence level of 95%. Regarding the scalability of the test loading, the loading times adjusted the model with a p-value of 0.0038. To be a relevant result, its p-value should be below 0.05 due to the confidence level of 95%. Therefore, it can be concluded that the method for loading tests scaled linearly since its results fit a linear regression model when the number of tests in the ATS grew.

Concerning the scalability of the test execution, the execution times for the tests with full execution adjusted the linear model with a p-value of 0.0121, while the execution times for the tests with undefined results adjusted the linear model with a p-value of 0.003. Since both p-values are lower than 0.05 and the confidence level is 95%, it can be concluded that the method scaled linearly for the test execution since its results fit a linear regression model when the number of tests grew. This linear behaviour can be observed thanks to the blue line depicted in Fig. 7, which corresponds to the regression line adjusted to the method results.

5.2. Minimum common knowledge between ontologies

In addition to the experiment related to the conformance of the VICINITY ontology network with regards to the IoT standards, it was also analysed the minimum common knowledge between them. The aim of this experiment was to identify the common knowledge shared by a set of standards to check whether there is an overlap in the domain they describe.

As for the previous experiment, the standards needed to have associated requirements that identify the knowledge defined in them or documentation that allows extracting the requirements. However, they do not need to have an ontology associated. As it was performed for the previous experiment, the tests associated with these requirements were defined following the testing process and the testing language described in Section 3. In this scenario, all the tests were also abstract, i.e., independent of the ontology from which they are extracted, and were executed on all the standards.

The following steps summarise the actions performed during the experiment:

1. To gather the functional requirements related to the standards.
2. To group them according to the topic associated, as described in Section 3.1.1.
3. To generate the ATS associated with these requirements following the conformance testing process and testing language described in Section 3.1.2.
4. To group the test designs in the ATS according to the type of TP and topic associated.
5. To execute all the ATSs of all the standards on all the standards.
6. To analyse the results to identify classes and properties that are shared between more than one standard.

The following information was retrieved from the performance of the previous steps:

- The requirements that are shared between more than one standard.

Table 11
Shared requirements between the IoT standards.

Provenance	Requirement	Type of requirement	Topic	Standards that satisfy the requirement
ISO/IEC 30141	Actuators and sensors are kinds of IoT device	Subsumption of classes	Device	ISO/IEC 30141, SAREF
OCF	A device is a logical entity	Definition of a class	Device	ISO/IEC 30141, OCF, oneM2M, SAREF
OCF	A device can be composed by other devices	Relation between classes	Device	OCF, oneM2M, SAREF
OCF	A resource is a physical thing in the world	Definition of a class	Thing	OCF, oneM2M
oneM2M	A controlling functionality represents a functionality that has impacts on the real world, but does not gather data	Subsumption between classes	Function	oneM2M, SAREF
oneM2M	A measuring functionality represents a functionality that has no impacts on the real world, but only gathers data	Subsumption between classes	Function	oneM2M, SAREF
oneM2M	A device can be composed of several (sub-)devices	Relation between classes	Device	oneM2M, SAREF
oneM2M	A thing is an entity that can be identified in the oneM2M System.	Definition of a class	Thing	ISO/IEC 30141, oneM2M
SAREF	A device shall have a model property	Relation between terms	Device, Property	ISO/IEC 30141, OCF, oneM2M, SAREF
SAREF	What is a device?	Definition of a class	Device	ISO/IEC 30141, OCF, oneM2M, SAREF
SAREF	A device can optionally have a description	Relation between terms	Device	ISO/IEC 30141, OCF, oneM2M, SAREF
SAREF	A device may consist of other devices	Relation between terms	Device	ISO/IEC 30141, oneM2M, SAREF
SAREF	A function represents the functionality necessary to accomplish the task for which a device is designed	Relation between terms	Device, Function	oneM2M, SAREF
SAREF	A service is a representation of a function to a network that makes this function discoverable; registerable and remotely controllable by other devices in the network	Relation between terms	Service, Function, Device	oneM2M, SAREF
SAREF	A device can be used for the purpose of sensing	Relation between terms	Device, Function	oneM2M, SAREF
SAREF	A device can be used for measuring a property	Relation between classes	Device, Function	oneM2M, SAREF
SAREF	A command can act upon a state to represent that the consequence of a command can be a change of state of the device	Definition of a class	Command	oneM2M, SAREF
SAREF	A device offers a service	Relation between terms	Device, Service	oneM2M, SAREF
SSN	What is a sensor?	Definition of a class	Sensor	ISO/IEC 30141, SAREF, SSN
SSN	What is an actuator?	Definition of a class	Actuator	ISO/IEC 30141, SAREF, SSN
SSN	What is a property?	Definition of a class	Property	OCF, SAREF, SSN

- The types of the requirements shared between more than one standard.
- The topics related to the requirements shared between more than one standard.
- The classes and properties that are shared between more than one standard. These results allow to identify the layers of knowledge described in Section 4, namely, the *common knowledge*, which represents the knowledge shared by the set of analysed standards, and *variant-domain knowledge*, which represents the knowledge that is common to more than one standard.

To execute the tests, the extension of the tool Themis that automatically generates ontologies from their associated ATS for those standards without a related ontology was used. This automatic generation of ontologies allows the execution of the tests on any standard.

For this experiment, the IoT standards used in the previous experiment, i.e., the ETSI SAREF ontology, the W3C SSN ontology, the oneM2M base ontology, the ISO/IEC 30141 and the OCF standard, were selected to be analysed. As mentioned in the previous section, neither the OCF standard nor the ISO/IEC 30141 have related ontologies.

5.2.1. Results

After executing all the tests on all the IoT standards, the test results analysis activity was performed. During this test results analysis, it was checked which requirements, and from which type and topic, are shared among the standards.

This analysis showed that none of the requirements and no terms were shared by all of the standards, i.e., the common knowledge. However, the variant-domain knowledge, which refers to the knowledge shared by more than one standard, could be identified. The results of such analysis are summarised in Table 11, where the ontologies that satisfy each requirement are indicated, as well as the type and topic associated with it. This table only includes those requirements that are shared by more than one ontology.

Table 11 shows that only 21 requirements out of 197, i.e., 10.65% of the requirements, are shared between the analysed standards. This confirms that only a small number of requirements are shared between standards and ontologies related to the same domain. These shared requirements represent the variant-domain knowledge related to the standards. Hence, hypothesis H1 is validated.

Moreover, from Table 11 it can be observed that the majority of these requirements are related to the definition of classes (e.g., the

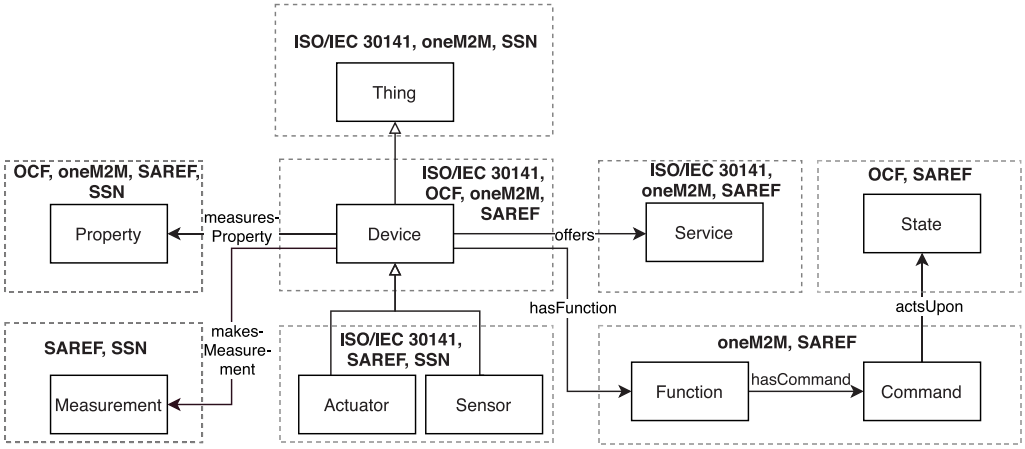


Fig. 8. Overview of the shared terms between standards.

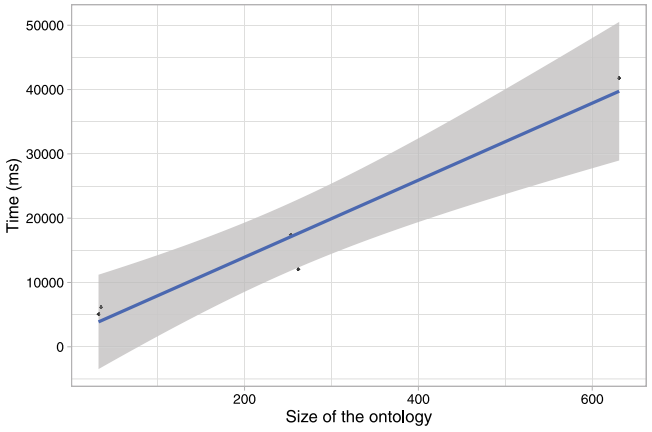


Fig. 9. Time spent during the execution of the ATSs on each standard.

Table 12
Terms shared by the analysed standards.

Term	Standards that contain the term
ActuatingFunction	oneM2M (as ControllingFunction), SAREF
Actuator	ISO/IEC 30141, SAREF, SSN
Command	oneM2M, SAREF
Device	ISO/IEC 30141, OCF, oneM2M, SAREF
Function	oneM2M, SAREF
Measurement	SAREF, SSN (as Observation)
Network	ISO/IEC 30141, SAREF
Property	OCF, oneM2M, SAREF, SSN
Service	ISO/IEC 30141, oneM2M, SAREF
Thing	ISO/IEC 30141, oneM2M, SSN (as Feature of Interest)
Sensor	ISO/IEC 30141, SAREF, SSN
State	OCF, SAREF,
SensingFunction	oneM2M (as MeasuringFunction), SAREF
ConsistsOf	OCF, SAREF, oneM2M
Hosts	OCF, SSN
hasCommand	oneM2M, SAREF
hasFunction	oneM2M, SAREF
hasInput	oneM2M, SSN
hasManufacturer	OCF, SAREF,
hasOutput	oneM2M, SSN
hasProperty	OCF, SSN
hasValue	oneM2M, SAREF
Offers	oneM2M, SAREF
Represents	ISO/IEC 30141, SAREF, oneM2M

requirement *"What is a sensor?"* only checks that there is a Sensor in the ontology) and relationships between them (e.g., the requirement *"A device offers a service"* checks the relation between the classes Device and Service). This table also shows that only 8 topics have shared requirements. The requirements related to the rest of topics are defined to describe the particular area of concern of the associated standard, e.g., Profile in the case of SAREF, Sample in the case of SSN or Operation in the case of oneM2M.

To check if there are more classes and properties shared by the standards, but not included in the requirements, the shared glossary of terms was analysed. Table 12 shows the ontologies that define each term included in such glossary. To improve the readability of the results, Table 12 only includes the information shared by more than one ontology.

From Tables 11 and 12 it can be observed that in this scenario the majority of the shared requirements are related to either the definition of classes or relations, while restrictions such as cardinalities are not shared. Joining these tables with the information retrieved from Table 9, where the most shared type of requirement is the one related to the definition of classes, hypothesis H2 is also validated. With both experiments, it was concluded that the variant-domain knowledge between the standards is related to the less restrictive requirements in the requirement specification documents, such as those related to the definition of classes or relations between them.

Fig. 8 shows the classes shared by more than one standard. From this figure it can be observed that some of the standards are related to generic terms, e.g., a thing, while other standards describe more specific aspects of the domain, e.g., commands associated to functions performed by devices.

Based on the results obtained after the application of the method in this use case, ontology developers and users can be aware of how these different standards cover the IoT domain, as well as of which are their overlaps and which are their differences. Developers and users can employ these results for checking that there is a consensus between these standards regarding the terms they describe and that there are no conflicts between them. Moreover, developers could also benefit from these results for selecting which standard better fits the developers' needs in ontology reuse tasks. As an example, SAREF and oneM2M can be useful for the definition of functions related to a device, while SAREF and SSN can be useful for the definition of measurements.

5.2.2. Performance

The time spent during the execution of all the test suites on each standard using Themis was calculated. Fig. 9 depicts the results and how the size of the standards, which is measured as the number of axioms in the ontology, influences the execution time. It is worth noting that in this scenario the same 197 tests are executed on each standard.

To check the scalability of the method, a linear regression model was also applied with a confidence level of 95%. As a result, the execution times adjusted the model with a p-value of 0.0042. As mentioned in Section 5.1.2, to be a relevant result, its p-value should be below 0.05 due to the confidence level of 95%. Therefore, it can be concluded that the method scaled linearly since its results fit a linear regression model when the size of the ontology grew. This linear behaviour can be observed thanks to the blue line depicted in Fig. 9, which corresponds to the regression line adjusted to the method results.

6. Conclusions and future work

The adoption of conformance techniques in the Ontology Engineering field could help to ensure the quality of ontologies by verifying whether they conform to well-known standards. Nowadays, standardisation bodies develop and publish standard ontologies and data models, but they do not provide any method or technique to analyse conformance. As it is recommended by the W3C (Dubost et al., 2005), every specification of a standard should specify the conformance clauses and provide a method to analyse them. To that end, this work describes the first steps towards the analysis of conformance for ontologies based on functional ontology requirements.

In this paper, a method for analysing conformance between an ontology and a standard is proposed, which can be used by either ontology developers or users for checking the conformance of an ontology with regards to a standard ontology or to a non-ontological standard data model. Moreover, this method allows identifying areas of conflict between an ontology and such standard, as well as the topics of the standard that are not covered by the ontology. In terms of ontology reuse, the conformance testing method could also be used by ontology developers for identifying potential mappings or terms for reuse between the ontology and the standard.

Furthermore, this paper also proposed a method for determining the minimum knowledge shared by a set of standard ontologies and data models. This method can be used by ontology developers and users to analyse how a set of standards covers a particular domain, checking whether there is a consensus concerning the definition of such domain and whether there are conflicts between them.

Both methods were applied to a particular use case, where the ontology network developed in the VICINITY project was analysed regarding its conformance with a set of standard ontologies related to the IoT domain.

The analysis of conformance in this use case shows how the shared requirements between the ontology network and the standards, as well as the variant-domain knowledge between standards in a particular domain are mostly related to the definition of classes. Those restrictive requirements, e.g., those related to cardinalities, are only satisfied by the ontology or standard from which they are extracted. These restrictive requirements refer to domain-task knowledge that is defined for particular aspects of the described domain.

Besides, it has also been observed that the analysed standards in the IoT domain (i.e., the SAREF ontology, the SSN ontology, the oneM2M ontology, the ISO/IEC 30141 and the OCF standard) do not share any minimum commitment between all of them, i.e., common knowledge, although some of them share some terms and requirements. These results show that even though these IoT standards are related to the same domain, they were created to provide support to different areas of concern in the same IoT field and, therefore, there is a minimum overlap between them.

During the validation of the conformance and the minimum common identification methods, it was observed that the results depend on the quality of the requirements specification. Therefore, to provide an accurate conformance analysis, standardisation bodies should spend effort in providing precise requirements specifications. Moreover, although the requirements specification should include all the different types of requirements, standardisation bodies should also consider including requirements related to the definition of classes in their specifications, since there is a small number of them included.

Future work will be directed to provide guidelines on how to extract requirements from standard specifications, as well as to study the automatic translation from ontology requirements into tests, to ease the conformance testing process. Moreover, future work will be also directed to the adoption of certification practices in the Ontology Engineering field, to propose techniques to issue certificates for ontologies based on their conformance with standards.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is partially supported by the H2020 project VICINITY: Open virtual neighbourhood network to connect intelligent buildings and smart objects (H2020-688467) and by a Predoctoral grant from the I+D+i program of the Universidad Politécnica de Madrid.

References

Baader, F., Calvanese, D., McGuinness, D., Patel-Schneider, P., Nardi, D., et al., 2003. The Description Logic Handbook: Theory, Implementation and Applications. Cambridge university press.

Baader, F., Horrocks, I., Sattler, U., 2004. Description logics. In: Handbook on Ontologies. Springer, pp. 3–28.

Beck, K., 2003. Test-Driven Development: By Example. Addison-Wesley Professional.

Bezerra, C., Santana, F., Freitas, F., 2014. CQChecker: A tool to check ontologies in OWL-DL using competency questions written in controlled natural language. Learn. Nonlinear Model. 12 (2), 115–129.

Blomqvist, E., Sepour, A.S., Presutti, V., 2012. Ontology testing-methodology and tool. In: Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2012), Galway City, Ireland, September 8-12, 2012, pp. 216–226.

Cuenca, J., Larrinaga, F., Curry, E., 2019. Experiences on applying SPL engineering techniques to design a (Re) usable ontology in the energy domain. In: Proceedings of of the 31st International Conference on Software Engineering and Knowledge Engineering (SEKE 2019), Lisbon, Portugal, July 10-12, 2019.

Dahlström, E., Dengler, P., Grasso, A., Lilley, C., mcCormack, C., Schepers, D., Watt, J., Ferraiolo, J., Jun, F., Jackson, D., 2011. Scalable Vector Graphics (SVG) 1.1 Specification. W3C.

Davies, K., Keet, C.M., Ławrynowicz, A., 2017. TDDonto2: A test-driven development plugin for arbitrary tbox and abox axioms. In: Proceedings of the 14th European Semantic Web Conference (ESWC 2017), Portoroz, Slovenia, May 28-June 1, 2017. Springer, pp. 120–125.

Dennis, M., van Deemter, K., Dell'Aglio, D., Pan, J.Z., 2017. Computing authoring tests from competency questions: Experimental validation. In: Proceedings of the 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 21-25, 2017, pp. 243–259.

Dubost, K., Rosenthal, L., Hazaël-Massieux, D., Henderson, L., 2005. QA Framework: specification guidelines. W3C Recomm..

Duque-Ramos, A., Fernández-Breis, J.T., Stevens, R., Aussenac-Gilles, N., et al., 2011. OQuaRE: A square-based approach for evaluating the quality of ontologies. J. Res. Pract. Inf. Technol. 43 (2), 159.

ETSI, 2017. ETSI TS 103 264 V2.1.1. SmartM2M; Smart Applications; Reference Ontology and oneM2M Mapping. Technical Report.

Fernández-Izquierdo, A., García-Castro, R., 2018. Requirements behaviour analysis for ontology testing. In: Proceedings of the 21st International Conference on Knowledge Engineering and Knowledge Management (EKAW 2018), Nancy, France, November 12-26, 2018. Springer, pp. 114–130.

Fernández-Izquierdo, A., García-Castro, R., 2019. Themis: A tool for validating ontologies. In: Proceedings of the 31st International Conference on Software Engineering and Knowledge Engineering (SEKE 2019), Lisbon, Portugal, July 10-12, 2019.

Fernández-Izquierdo, A., Poveda-Villalón, M., García-Castro, R., 2019. CORAL: a corpus of ontological requirements annotated with lexico-syntactic patterns. In: Proceedings of the 16th European Semantic Web Conference (ESWC 2019), Portoroz, Slovenia, June 2-6, 2019. Springer, pp. 443–458.

Gangemi, A., Presutti, V., 2009. Ontology design patterns. In: Handbook on Ontologies. Springer, pp. 221–243.

García-Castro, R., 2009. Benchmarking Semantic Web Technology, Vol. 3. IOS Press.

Grabowski, J., Hogrefe, D., Réthy, G., Schieferdecker, I., Wiles, A., Willcock, C., 2003. An introduction to the testing and test control notation (TTCN-3). Comput. Netw. 42 (3), 375–403.

Graydon, P., Habli, I., Hawkins, R., Kelly, T., Knight, J., 2012. Arguing conformance. IEEE Softw. 29 (3), 50–57.

Grüninger, M., Fox, M.S., 1995. Methodology for the Design and Evaluation of Ontologies. Citeseer.

Guarino, N., Welty, C., 2002. Evaluating ontological decisions with ontoclean. Commun. ACM 45 (2), 61–65.

Haller, A., Janowicz, K.V., Cox, S., Le Phuoc, Lefrançois, M., Atkinson, R., García-Castro, R., Lieberman, J., Stadler, C., 2017. Semantic sensor network ontology. W3C Recomm..

Haller, A., Janowicz, K., Cox, S.J., Lefrançois, M., Taylor, K., Le Phuoc, D., Lieberman, J., García-Castro, R., Atkinson, R., Stadler, C., 2019. The modular SSN ontology: A joint w3c and OGC standard specifying the semantics of sensors, observations, sampling, and actuation. Semant. Web 10 (1), 9–32.

Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S., et al., 2009. OWL 2 web ontology language primer. W3C Recomm..

Horridge, M., Patel-Schneider, P.F., 2012. OWL 2 Web Ontology Language Manchester Syntax. W3C Working Group Note.

2017. ISO/IEC 30141:2017: Internet of Things (IoT) - Reference Architectures. International Organization for Standardization, Geneva, Switzerland.

1994. ISO/IEC 9646-1: Information Technology – Open Systems Interconnection – Conformance Testing Methodology and Framework, Vol. 1. International Organization for Standardization.

Kaebisch, S., Kamiya, T., McCool, M., Charpenay, V., Kovatsch, M., 2020. Web of things (WoT) thing description. W3C Recomm..

Keet, C.M., Ławrynowicz, A., 2016. Test-driven development of ontologies. In: Proceedings of the 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 17-21, 2016, pp. 642–657.

Kristoffersen, F., Walter, T., 1996. TTCN: Towards a formal semantics and validation of test suites. Comput. Netw. ISDN Syst. 29 (1), 15–47.

Lawrynowicz, A., Keet, C.M., 2016. The TDDonto tool for test-driven development of DL knowledge bases. In: Proceedings of the 29th International Workshop on Description Logics (DL 2016), Cape Town, South Africa, April 22-25, 2016.

Lozano-Tello, A., Gómez-Pérez, A., 2004. Ontometric: A method to choose the appropriate ontology. J. Database Manag. 15 (2), 1–18.

Moseley, S., Randall, S., Wiles, A., 2003. Experience within ETSI of the combined roles of conformance testing and interoperability testing. In: Proceedings of the 33rd Conference on European Solid-State Device Research (ESSDERC 2003), Estoril, Portugal, September 16-18, 2003. IEEE, pp. 177–189.

Muggeo, V.M., 2008. Segmented: an r package to fit regression models with broken-line relationships. R news 8 (1), 20–25.

oneM2M, 2016. TS-0012 oneM2M Base Ontology. Technical Report.

Poveda-Villalón, M., Gómez-Pérez, A., Suárez-Figueroa, M.C., 2014. OOPS! (ontology pitfall scanner!): An on-line tool for ontology evaluation. Int. J. Semant. Web Inf. Syst. 10 (2), 7–34.

Prud'hommeaux, E., Seaborne, A., 2008. Sparql query language for rdf. W3C Recomm..

Roth, R.R., 2013. Keyword based software testing system and method. US Patent 8, 522, 214.

Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernández-López, M., 2012. The NeOn methodology for ontology engineering. In: Ontology Engineering in a Networked World. pp. 9–34.

Suárez-Figueroa, M.C., Gómez-Pérez, A., Villazón-Terrazas, B., 2009. How to write and use the ontology requirements specification document. In: Proceedings of the 8th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE 2009), Vilamoura, Portugal, November 3-4, 2009. Springer, pp. 966–982.