The Journal of Systems and Software 86 (2013) 2877–2889

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

# A systematic review on the functional testing of semantic web services

Abbas Tahir*,[1], Davide Tosi, Sandro Morasca

*Department of Theoretical and Applied Sciences, Università degli Studi dell'Insubria, Via Mazzini 5, I-21100 Como, Italy*

ARTICLE INFO

*Article history:*
Received 13 June 2012
Received in revised form 21 March 2013
Accepted 30 June 2013
Available online 9 July 2013

*Keywords:*
Functional testing
Semantic web services
Testing approach
Systematic literature review

ABSTRACT

Semantic web services are gaining more attention as an important element of the emerging semantic web. Therefore, testing semantic web services is becoming a key concern as an essential quality assurance measure. The objective of this systematic literature review is to summarize the current state of the art of functional testing of semantic web services by providing answers to a set of research questions. The review follows a predefined procedure that involves automatically searching 5 well-known digital libraries. After applying the selection criteria to the results, a total of 34 studies were identified as relevant. Required information was extracted from the studies and summarized. Our systematic literature review identified some approaches available for deriving test cases from the specifications of semantic web services. However, many of the approaches are either not validated or the validation done lacks credibility. We believe that a substantial amount of work remains to be done to improve the current state of research in the area of testing semantic web services.



## 1. Introduction

Software testing plays a prominent role in the assessment and improvement of software quality, which is an essential issue in software development. IEEE Std. 610.12-1990 (IEEE, 1990) defines testing as “the process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component.” One of the main goals of testing is to trigger failures, so that, based on the occurrence and nature of the failures, software developers are guided in the identification and removal of faults. Although there is wide agreement on its importance for software quality assurance, testing is often not performed systematically enough. One possible explanation is that testing is a cost- and time-intensive process. Pezzè and Young (2007) reported that “The cost of software verification often exceeds half the overall cost of software development and maintenance.” Therefore, testing techniques that help increase the efficiency of the testing activities could be very useful.

Recently, web services have become more and more attractive as a new paradigm for building software. W3C (2004) defines a web service as “a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL[2]). Other systems interact with the web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards.” Web services offer many benefits, like assured interoperability, modularity, and reusability. Conversely, they introduce a number of challenges and concerns. One of these challenges is their quality, seen from the perspectives of two stakeholders: (1) the developer/provider of web services and (2) the consumer of web services. For the developer, the main concern is that the web service is implemented as specified. Thus, the developer needs to take any reasonable measure to make sure that a web service is implemented correctly, including testing it. The web service consumer is mainly interested in whether the web service is the right service to use. It is therefore important to test web services as a quality assurance measure from the consumer's perspective as well.

Web services are usually described only syntactically, so only the structure of the data is specified, but not the semantics. This introduces a set of problems (such as integration inconsistencies (Denaro et al., 2009)) that can be partially addressed by the adoption of semantic web services, which support the semantic description of their behavior. In the semantic web service paradigm, data become machine-readable and understandable. Semantic web services can dynamically collaborate in processing data without losing their meaning. Ontology description languages (e.g., OWL-S (W3C, 2004), WSMO (WSMO, 2004) and WSDL-S (W3C, 2005))

* Corresponding author. Tel.: +49 89 63642750; fax: +39 031 2386309.
*E-mail addresses:* tahir_a@acm.org, abbas.a.tahir@googlemail.com (A. Tahir), davide.tosi@uninsubria.it (D. Tosi), sandro.morasca@uninsubria.it (S. Morasca).
[1] Abbas Tahir is also with Siemens AG, Corporate Technology, Otto-Hahn-Ring 6, 81739 Munich, Germany.

[2] A list that covers all relevant acronyms is provided in Table A.1 in Appendix A.

0164-1212/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jss.2013.06.064

are typically used to describe web services semantically. Characteristics of semantic web services, such as dynamic service composition, raise more testing challenges compared to the syntactically described web services. The semantical layer, is one of the main differences between semantic web services and traditional ones. Testing needs to be performed over the semantic layer and not through the lower syntactic layer as when testing traditional web services.

This systematic literature review aims to identify possible challenges associated with testing semantic web services. Two main differences between traditional and semantic web services may influence the approaches to be followed when testing semantic web services:

- *The presence of the semantic layer*: Semantics of traditional web services are not specified. The tester (or human user) of traditional web services need to make her/his own interpretation of the following:
  - The capability of the only syntactically specified functionality of the service (i.e., which goals the service can fulfill).
  - The preconditions need to be fulfilled before invoking the service.
  - The state changes resulting from the execution of the service.

  In the case of semantic web services, the above semantic information is pre-specified by means of ontology languages such as WSMO or OWL-S. The specified semantic information is machine-readable and understandable. Therefore, this semantic information can be used to guide testers when specifying test cases.
- *Heterogeneity of standards*: There are currently different initiatives and different non-compatible ontology languages (WSMO, OWL-S, WSDL-S, etc.) for semantic web services that exhibit different levels of formality. The more formal the description of the semantic web service, the more precise and comprehensive the testing can be (e.g., by following formal testing methods). This may also imply the use of different testing approaches depending on the level of formality of the semantic web services description models.

We carried out this systematic literature review according to the procedure outlined in Kitchenham and Charters (2007). Systematic literature reviews are widely used by researchers in the medical domain (e.g., the Cochrane reviews (Alderson et al., 2003)) and were recently adopted in software engineering research (Kitchenham, 2004). Systematic literature review is a form of secondary study that uses a well-defined methodology to identify, analyze, and interpret all available evidence related to a specific research question in a way that is unbiased and repeatable (Kitchenham and Charters, 2007) (to a degree). A systematic literature review (Kitchenham, 2004) is "a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest. Individual studies contributing to a systematic review are called primary studies."

The objective of this systematic literature review is to summarize the current state of the art of functional testing of semantic web services. Systematic literature reviews aim to focus on a clearly defined review topic. The topic selected for this systematic literature review focuses on functional testing of semantic web services, and is motivated by the following:

- Semantic web services is a new technology that just appeared a few years ago ((DAML, 2000)) and it is a very fervid research area, while traditional web services are already used in the industrial domain. Moreover, semantic and traditional web services are inherently different.
- We focused on functional testing because functionality is the primary quality of any software system and semantic web services are no exception. Additionally, functional testing is a widely used quality assurance technique.

Functional testing is concerned with testing the functionalities exposed by the semantic web service based on its functional specifications. Therefore, we intentionally excluded studies not related to functional testing and focused our analysis on studies addressing functional testing. The consumers and testers of the web services are not their developers in most cases. Therefore, the testers usually have access to the web service only through its specified and advertised interface without knowing the internal-working of the service. As a result, a web service can only be tested as a black-box. We are planning to focus on other quality assurance techniques like, static analysis, review, non-functional testing, etc. in future systematic literature reviews.

Our review follows the procedure[3] outlined by Kitchenham and Charters (2007). Thus, we identified a set of research questions derived from the objective of this review. Based on the research questions and the relevant search terms, we searched 5 well-known digital libraries, thus identifying 425 relevant primary studies. We then followed a rigorous selection procedure to filter the search results according to predefined criteria. The process resulted in the selection of 34 primary studies. We collected relevant data from the selected primary studies and then analyzed and summarized the data in different forms to provide answers to three research questions on semantic web services that are described in Section 3.1.

Other reviews were previously conducted in related areas like software product line testing (Engström and Runeson, 2011), web service composition testing (Rusli et al., 2011) and regression test selection techniques (Engström et al., 2010). We could identify three reviews that appear to be more relevant to the topic of our systematic review than the others. In their review, Palacios et al. (2011) focused mainly on service-oriented architectures where the discovery and binding of the services are performed at runtime. They analyzed testing approaches, the stakeholders involved in the testing effort, and the point of time the test is done (i.e., before service publication, during execution, etc.). Semantic web services addressed in this study and web services in general are usually considered as the main building blocks of service-oriented architectures. A much wider scope was considered by Bozkurt et al. (2012) in their survey. They covered testing and verification in service-oriented architectures in general without restricting themselves to services with dynamic binding. Testing of both functional and non-functional properties was considered. Moreover, different testing techniques available in the surveyed literature were presented and discussed. Zakaria et al. (2009) presented a review on unit testing approaches for BPEL (OASIS, 2007). They identified, categorized, and analyzed different BPEL unit testing approaches. BPEL describes interactions between web services. Although both Palacios et al. (2011) and Zakaria et al. (2009) considered the identification and the description of available testing approaches (as we do), both reviews focus on different and very specific subjects. Bozkurt et al. (2012) has a much wider scope that encompasses testing service-oriented architectures in general. Our systematic literature review focuses on testing a special type of web services: semantic web services.

The remainder of this paper is organized as follows. Section 2 provides background information on two leading semantic web

[3] We also used the same terminology used by Kitchenham and Charters (2007) throughout this article.

services intiatives. In Section 3, we describe the research methodology in terms of the research questions to be answered and the methodology followed to provide reliable answers for them. Section 4 reports the results of this study and provides answers to the research questions based on our systematic review. In Section 5, the results are discussed and directions for future research are proposed. Section 6 discusses possible threats to the validity of this study. Finally, in Section 7, we provide our conclusions.

## 2. Background

There are two major initiatives in the area of semantic web services, namely WSMO (WSMO, 2004) and OWL-S (W3C, 2004). WSMO is a conceptual model for semantic web services. WSMO web services are described explicitly in terms of their functional and non-functional properties and their interfaces using WSMO in a way that allows for automatic discovery, selection, composition, mediation and execution of the web services. The main elements of a WSMO web services are:

- *Capability*, which describes the functionality of the web service,
- *Interfaces*, which describes how the web service achieves its capability by means of interactions with its users (Choreography) and by using other web services (Orchestration).

WSMO follows a goal-based approach in which the user defines her/his goals by explicitly expressing her/his requirements on the WSMO web services. Based on user goals, the WSMO framework discovers the suitable services by automatically matching user goals to the semantically described capabilities of the published web services. If necessary, the framework uses Mediators to handle interoperability problems or Orchestration that automatically combines services based on their capabilities to satisfy user goals.

Similar to WSMO, OWL-S provides another specification to describe web services semantically. OWL-S web services are described in terms of three elements, as follows.

- *Service Model* mainly describes the control flow of the service (*Process Model*). Additionally, it provides a process for the automatic composition and invocation of web services.
- *Service Profile* describes the capabilities of the service including a functionality and quality of service parameters.
- *Service Grounding* provides a mapping of the semantical description to the syntactically described implementations of web services in WSDL. It also specifies communication protocols, transport mechanisms, and message format.

OWL-S web services are discovered by referencing their capabilities described in the *Service Profiles*. If no single service matches the requested user service, the *Process Model* describes how an automatic composition of different services based on their semantics can be achieved to satisfy the request.

## 3. Research method

This systematic review was conducted following the procedure outlined by Kitchenham and Charters (2007). We followed the special recommendations for PhD students[4] where applicable.

[4] One of the authors (Abbas Tahir) is a PhD student.

### 3.1. Research questions

Defining research questions is an essential part of a systematic review, as they drive the entire review methodology (Kitchenham and Charters, 2007). To achieve the objectives of this review, we identified the following research questions.

RQ1. Is it possible to derive functional test cases from requirement specifications of semantic web services? What approaches are used?

RQ2. What are the challenges associated with the derivation of test cases from the specifications of semantic web services?

### 3.2. Search strategy

As a necessary starting point, systematic literature reviews aim to find all primary studies related to the research questions in focus. To search for and find relevant studies, we need first to identify relevant search terms. For this purpose, we followed the approach outlined by Kitchenham and Charters (2007) in which we consider the research questions from three viewpoints: population, intervention, and outcomes. For each viewpoint, the relevant search terms in the context of this systematic literature review were identified as follows.

- **Population**: semantic web services, OWL-S, WSMO, WSDL-S.
- **Intervention**: test generation, test, testing, verification, validation, test case.
- **Outcomes**: functional properties.

In addition to these search terms, we considered synonyms, abbreviations, and alternative spellings to construct a search string. The search string was constructed as follows:

$$(P_1 \text{ OR } P_2 \ldots \text{OR } P_n) \text{ AND } (I_1 \text{ OR } I_2 \ldots \text{OR } I_n)$$

where $P_n$ refer to population terms, $I_n$ refer to intervention terms connected using the Boolean operators AND and OR. Purposely, we did not include the outcomes in the search string to broaden the scope of the results. Preliminary searches, which we conducted to assess the volume of potentially relevant studies, confirmed that the inclusion of outcomes in the search string remarkably reduces the volume of identified relevant studies and hence increases the risk of missing relevant studies. To compensate for not including outcomes in the search string, we filtered any study that does not address functional testing in the study selection process.

The search string was used to search the following five well-known and widely used digital libraries:

- ACM Digital library. [5]
- IEEE Xplore. [6]
- Inspec. [7]
- ScienceDirect. [8]
- SpringerLink. [9]

It was necessary to adjust the search string according to the requirements of each digital library. The search string was preliminarily checked against a list of already known primary studies

[5] http://dl.acm.org.
[6] http://ieeexplore.ieee.org.
[7] http://www.theiet.org/publishing/inspec.
[8] http://www.sciencedirect.com.
[9] http://www.springerlink.com.

((Shaban et al., 2009; Paradkar et al., 2007; Li et al., 2010; Wen-Jie and Shi, 2009)) as recommended by Kitchenham and Charters (2007). This preliminary check was used to examine the effectiveness of the search string before conducting the full search. The search was conducted in March[10] 2012 and was limited to studies published between the year 2000 and 2011. We decided to choose 2000 as the starting year for the search since the first ontology language for the web (DAML+OIL) was first introduced by the DAML project[11] at the end of that year (DAML, 2000).

### 3.3. Study selection process

Fig. 1 depicts the search stages followed and the resulting number of primary studies for each stage. In stage 1, automated search was performed by applying the search string to the digital libraries. The search was conducted on titles and abstracts of the studies. We obtained 425 studies, many of which were irrelevant, because the search was carried out only electronically. After that, in stage 2, duplicates were identified and removed. In stage 3, studies were excluded based on the title and the language according to the following two criteria:

1 Studies that do not address functional testing of semantic web services.
2 Studies that are not in the English language.

In stage 4, only the first criterion was applied to exclude studies after studying their abstracts. After stage 4, we checked titles of references in the 34 studies selected to identify any relevant primary studies to be included.

To assess the reliability of the study inclusion/exclusion criteria, a re-evaluation of a random sample of the primary studies was performed. The re-evaluation included checking the consistency of the inclusion/exclusion decisions made. The re-evaluation was done after stage 3 and stage 4 of the selection procedure. After stage 3, we selected randomly 4 included studies (i.e., fulfilled the selection criteria of this stage) and another 4 excluded studies. We re-checked each of the 8 studies by applying the inclusion/exclusion criteria of stage 3 to them again. The same re-evaluation process was applied to another 8 randomly selected studies form stage 4.

### 3.4. Study quality assessment

Kitchenham and Charters (2007) insist on the quality assessment of the primary studies, to minimize bias and maximize validity when evaluating the primary studies. They also list five different purposes for the assessment. In this work, we use the study quality assessment as a means of weighting the importance of individual studies when results are being synthesized.

We developed a study quality questionnaire composed of 6 questions inspired to the questions presented by Dyba et al. (2007) and Kitchenham and Charters (2007). The following questions were used for the assessment of quality of the primary studies:

QA1. Are the aims and the objectives of the primary study clearly reported?
QA2. Is the context in which the research was carried out adequately described?
QA3. Is the test case derivation technique (RQ1) presented in the study clearly described?
QA4. Is there any credible validation of the technique/approach?
QA5. Are the findings clearly stated and related to the goals of the study?
QA6. Do the conclusions relate to the defined aim and purpose of study?

### 3.5. Data extraction

The data extraction phase involves collecting information relevant to the research questions from the primary studies selected. We have designed a data extraction form for this purpose (Table 1). We used the test-retest process (Kitchenham and Charters, 2007) for the purpose of checking the consistency and accuracy of the extracted data with respect to the original sources. After finishing the extraction of information for all selected studies, we randomly selected 3 primary studies and performed a second extraction of the data. We noticed only one little inconsistency for one primary study (Dong, 2009) where the test tool used was missing in the initial data extraction form.

## 4. Systematic literature review execution and results

### 4.1. Primary studies

Searching the electronic databases listed in Section 3.2 resulted in 425 relevant primary studies. The bibliography reference management tool JabRef (JabRef, 2011) was used to manage the references to the primary studies identified. The study selection procedure described in Section 3.3 was then applied to the studies. First, a two-phase duplicate identification and removal process was followed. In the first phase, duplicate studies were identified automatically using the duplicate identification capability of JabRef. The tool identified 15 duplicates. In the second phase of duplicate identification process, we searched for duplicates manually. In this phase 131 duplicate studies were identified. The large inconsistency between the numbers of duplicates detected automatically and those identified manually may be due to the low sensitivity of the duplicate detection algorithm utilized by JabRef. After duplicates removal only 279 primary studies remained. Afterward, the title of each study was reviewed and studies that clearly do not address functional testing of semantic web services or that are not in the English language were excluded. For example, the search result included a study in veterinary genetics (Hull et al., 2008) titled "Development of 37 microsatellite loci for the great gray owl (Strix nebulosa) and other Strix spp. owls" which is clearly out of the scope of this review. The title and language based exclusion resulted in excluding additional 191 studies with only 88 studies remaining. Subsequently, the abstract of each study was thoroughly reviewed and studies that do not address functional or self-adaptive testing of semantic web services were filtered out. As a result, only 34 primary studies remained, listed in Table A.2 in Appendix A (Askarunisa et al., 2010, 2011; Bai and Kenett, 2009; Bai et al., 2008, 2007; Chen et al., 2009; Dai et al., 2007; Dong, 2009; Goli and Pathari, 2006; Jokhio et al., 2009; Lee et al., 2008; Li et al., 2010; Liu et al., 2011; Ma et al., 2010; Noikajana and Suwannasart, 2008, 2009, 2008b; Oghabi et al., 2011; Paradkar et al., 2007; Park et al., 2009; Ramollari et al., 2009; Shaban et al., 2009; Sinha and Paradkar, 2006; Wang et al., 2009a,b, 2007; Wang and Sun, 2009; Wang and Huang, 2008; Wen-Jie and Shi, 2009; Yu and Li, 2010; Yu et al., 2007; Zakaria et al., 2009; Zhang et al., 2011; Zhang and Zhu, 2008). Fig. 2 shows the accumulated number of selected primary studies published from the year 2000 to the year 2011. The accumulated number of publications was increasing over the years with the largest increase in the year 2009 (13 Studies). The following years (2010 and 2011) witnessed a steady increase of 4 publications a year.

[10] We checked again in late October 2012 and we could not identify any significant new publications.
[11] http://www.daml.org/.

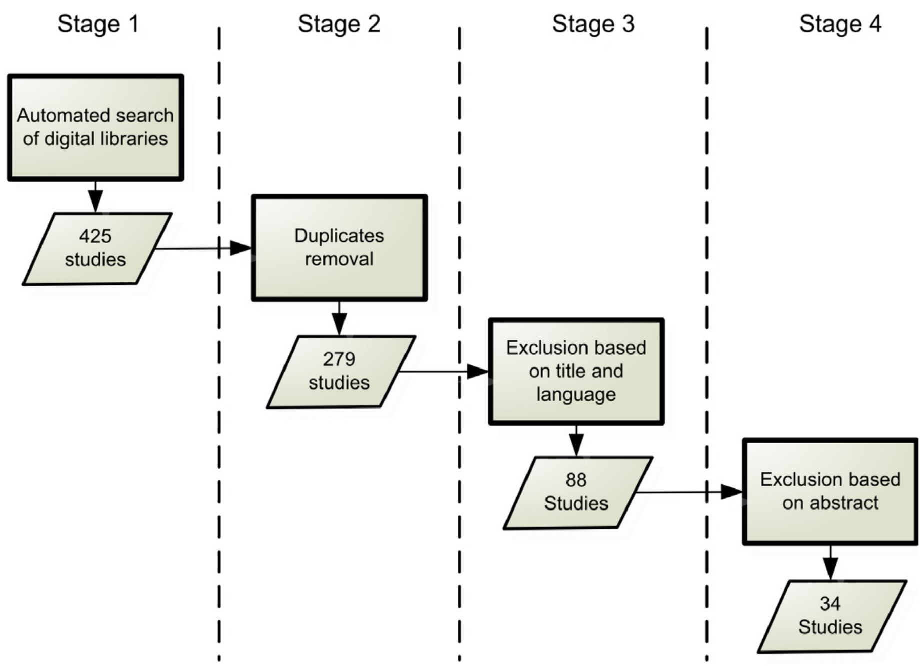


**Fig. 1.** Stages of the search strategy.

**Table 1**
Data extraction form.

| No. | Data extraction category | Description | Addresses |
|---|---|---|---|
| *General description* | | | |
| 1 | Identifier | Unique identifier of the primary study | |
| 2 | Date | Date of data extraction | |
| *Study description* | | | |
| 3 | Title | Title of the primary study | |
| 4 | Authors | Authors of the primary study | |
| 5 | Publication year | Publication year | |
| 6 | Type | Conference paper, Journal article, book chapter, workshop paper | |
| 7 | Publication medium | Name of the publishing conference, journal, book or workshop (Including, e.g., volume, number, page...) | |
| *Study contents* | | | |
| 8 | Objectives | What are the main objectives of the study? | RQ1, RQ2 |
| 9 | Testing technique | What is the testing technique utilized? | RQ1, RQ2 |
| 10 | How are the test cases derived | What is the base used for deriving test cases? | RQ1 |
| 11 | Challenges identified | What challenges are associated with the chosen test derivation approach? | RQ2 |
| 12 | Tool support/automation | Is the approach described in 10 or 12 supported by a tool or fully/partially automated? | RQ1 |
| 13 | Validation | which method is used for validating the study? | RQ1, RQ2 |

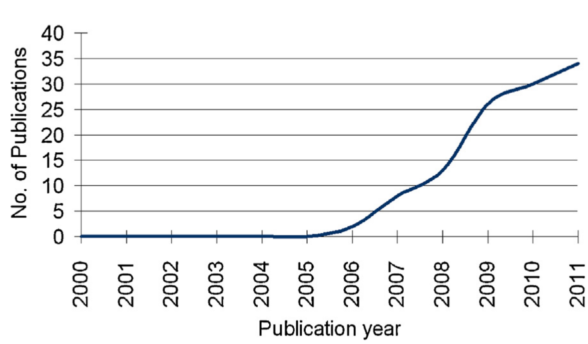


**Fig. 2.** Accumulated number of publications.

For the 34 studies selected, we conducted a full text review including reviewing the references list of each study looking for relevant ones. We basically checked the titles of the studies in the reference lists for relevance to the purpose of this systematic review. All references that we could identify as relevant were found already included in the 34 studies identified previously as in Table A.2 in Appendix A. This provided additional confidence on the effectiveness of the search process we followed.

As we mentioned in Section 3.4, it is essential to assess the quality of the primary studies selected. The assessment is used to weigh the importance of individual studies when results are being summarized (Kitchenham and Charters, 2007). We used a simple scale of two values: (*Yes*) and (*No*) to answer the quality assessment questions as in Table A.3 in Appendix A. Most of the questions received a positive answer (*Yes*). In what follows, we discuss only the few cases where the questions were answered negatively (*No*). For five studies (PS21, PS26, PS30, PS32, and PS33) the answer for question QA3 was negative. Specifically, studies PS21, PS26, PS32, and PS33 do not present any specific test case generation or test case derivation approach. Study PS21 presents a testing architecture based on SOA whereas study PS26 introduces a risk-based test case selection technique. Both PS32 and PS33 present test prioritization techniques. In PS30, the utilized test case derivation technique is insufficiently described because of space limitations.

For eight studies (PS1, PS11, PS20, PS21, PS24, PS26, PS28, and PS31), the answer to question QA4 was negative. Studies PS1, PS11, PS20, PS24, PS26, PS28 and PS31 do not convey any credible validation. In PS21 there is no specific test case generation or test case derivation approach presented and therefore no validation as required by QA4 is carried out. Study PS27 does not dedicate a separate section for the conclusions and in general does not discuss the conclusions.

### 4.2. Approaches for deriving functional test cases from requirement specifications (RQ1)

The central question in this systematic literature review, RQ1, focuses on the approaches available for deriving functional test cases from requirements specifications of semantic web services. Functional capabilities (or requirements) of the semantic web services are usually expressed by using an ontology description language (e.g., OWL-S, WSDL-S, etc.). Test cases can be derived from the requirements specifications of the semantic web services to test the degree to which these requirements are satisfied. Thirty primary studies selected describe approaches for testing semantic web services including the description of the test case derivation technique used. The primary study PS9 presents an automatic testing-based approach for the discovery of web services based on the semantic information provided by the service requester. Additionally, it proposes exploratory testing as a test case derivation technique when the semantic web service specifications are missing or not sufficiently documented. So, the aim of PS9 is not to introduce a new testing approach for semantic web services, but to utilize testing as a way to select a specific web service during discovery to satisfy a specific request based on the provided semantic information. Therefore, PS9 is out of the scope of RQ1. Only 3 primary studies (PS19, PS21 and PS26) do not present any testing approach. The objective of PS19 is to introduce a model for reasoning about the evolution of OWL-S requirements specification using $\pi$-calculus. PS21 presents an architecture for testing web services. Ontologies are adapted to describe testing concepts and relations, based on which the interoperation between services are specified and implemented in semantic web services technology. The presented architecture supports the dynamic discovery and invocation of testing services. Thus the study fullfils the search criteria but it does not provide any information about test case generation or derivation. PS26 introduces risk-based adaptive group testing for testing complex systems of semantic web services. Test cases are categorized and scheduled according to the risks associated with the targeted web services. The approach presented is not about test case generation or derivation but about a strategy for selecting a subset of test cases based on the risks associated with the web services. So, the study addresses functional testing of semantic web services, but it does not describe any test case generation or derivation approach.

Table 2 lists selected primary studies with test derivation base specifications (discussed in Section 4.2.1), transformations needed to any target model (discussed in Section 4.2.2), and the techniques followed to derive test cases from the specifications (discussed in Section 4.2.3).

#### 4.2.1. Test case derivation base specifications

All selected primary studies (Table A.2 in Appendix A) rely on some kind of specification models as a base for test case derivation (Table 2). The only exceptions are PS9, PS19, PS21, and PS26, which, as we mentioned before, do not present any testing approach.

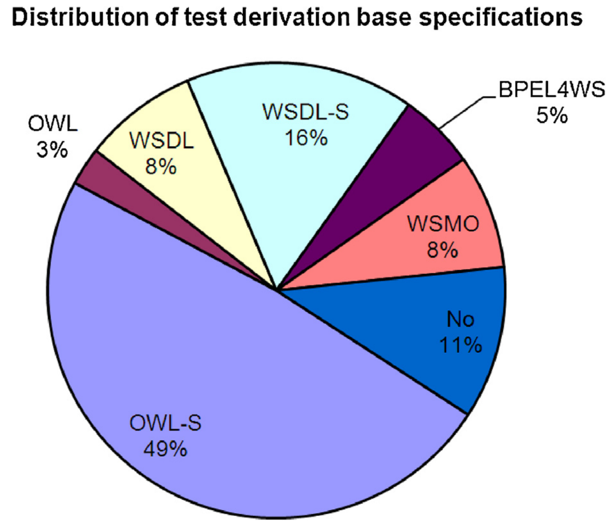


**Fig. 3.** Distribution of test case derivation base specifications.

Fig. 3 illustrates a pie chart of the distribution of base specifications used for test case derivation presented in the selected primary studies. Eighteen of the test approaches (49%) utilize OWL-S specifications models or OWL-S models augmented with a rule-based models (SWRL, RIF-PRD) and FTLTL (Future Time Linear Temporal Logic). All test approaches (other than PS5, PS9, PS14-1,[12] PS14-4, PS19, PS21, PS26, PS27, and PS32) rely on ontology-based specifications that describe the semantics of web services, via OWL-S, OWL, WSMO and WSDL-S. PS9, PS19, PS21, and PS26 do not introduce any testing approach and therefore are not relevant. PS5 and PS14-1 use a WSDL syntactical description model. In PS14-4 and PS27, BPEL4WS specification models are used. BPEL4WS is an abbreviation of Business Process Execution Language for Web Services. Although it can be used to semantically describe a web service (Mandell and McIlraith, 2003; Grigorova, 2006), BPEL4WS is basically not a semantic description language. In addition to WSDL specification, PS32 uses the IOPE (Input, Output, Precondition, and Effect) information without indicating the source of this information since such information cannot be described using WSDL.

#### 4.2.2. Model transformations

To simplify the automation of test case generation, it may be useful to represent the provided web service specifications in another format via model transformations. The original web service specification model is transformed into another target model that is believed to be much more efficient in terms of automatic test case generation or that is well supported by test generation tools. In the selected primary studies, the original specification model is transformed into one of the following models:

- Petri Net.
- B model.
- Promela.
- WSDL-S.
- IPM (Input Parameter Model).
- Fault model.

[12] PS14 represent four different techniques for testing web services. We added a suffix to the end of the primary study ID to differentiate from the test approaches presented, i.e., PS14-1, PS14-2. PS14-3 and PS14-4. PS14-1 uses WSDL specifications which do not include any semantic information. We included this approach in our systematic review for the sake of completeness.

**Table 2**
Approaches for deriving functional test cases from requirements specification of semantic web services.

| Primary study ID | Test case derivation base specifications | Model transformation into | Test case derivation technique |
|---|---|---|---|
| PS1 | OWL-S specifications | Petri Net model | Path traversing and Reasoning over IOPE |
| PS2 | WSMO specifications | B model | Model checking |
| PS3 | OWL-S specifications | Promela model | Model checking |
| PS4 | OWL-S and SWRL specifications | No | Reasoning |
| PS5 | WSDL and OCL specifications | WSDL-S specifications | Pair-wise testing and orthogonal array testing |
| PS6 | WSDL-S and SWRL specifications | No | Random testing |
| PS7 | WSDL-S and OCL specifications | Input Parameter Model (IPM) | Pair-wise testing |
| PS8 | OWL-S specifications | Fault models (testing goals) | Extended graph plan planning algorithm |
| PS9 | No | No | Exploratory testing |
| PS10 | OWL-S specifications | No | Mutation testing |
| PS11 | OWL-S specifications | High-level Petri Net (HPN) model | Model checking |
| PS12 | OWL-S specifications | Petri Net model | Path traversing &Reasoning over IOPE |
| PS13 | OWL-S specifications | Enhanced Hierarchical Color Petri Net (EH-CPN) model | Path traversing and partition testing |
| PS14-1 | WSDL specifications | No | Random testing |
| PS14-2 | WSDL-S specifications | No | Input and precondition analysis |
| PS14-3 | OWL-S specifications | No | Random testing |
| PS14-4 | BPEL4WS specifications | Petri Net model | Path traversing |
| PS15 | WSMO specifications | B model | Model checking |
| PS16 | OWL &RIF-PRD Specifications | Stream X-machine Model | W-method |
| PS17 | OWL-S specifications | Temporal Logic Actions (TLA) model | Model checking |
| PS18 | WSDL-S specifications | Extended Finite State Machine (EFSM) model | One of the following: (a) Full predicate coverage (b) BZ-TT method (c) Mutation based (d) User defined test objectives |
| PS19 | No | No | No |
| PS20 | OWL-S specifications | No | Mutation testing |
| PS21 | No | No | No |
| PS22 | OWL-S specifications | Petri Net model | Path traversing &Reasoning over IOPE |
| PS23 | OWL-S specifications | No | Partition testing |
| PS24 | OWL-S, FTLTL and SWRL specifications | No | Runtime analysis (code instrumentation) |
| PS25 | OWL-S specifications (extended to support the specification of mutant operators) | No | Mutation testing |
| PS26 | No | No | No |
| PS27 | BPEL4WS specifications | Labeled Transition System (LTS) | Condition checking |
| PS28 | WSMO specifications | No | Boundary conditions and equivalence classes testing |
| PS29 | WSDL-S &SWRL specifications | No | Decision tables testing |
| PS30 | WSDL-S &SWRL specifications and Stream X-machine Model | No clear description | |
| PS31 | OWL-S specifications | Flow graph-based test model | Path traversing &and other traditional white-box testing techniques |
| PS32 | WSDL specifications and IOPE information | No | No |
| PS33 | OWL-S specifications | No | No |
| PS34 | OWL-S specifications | Markov chain diagram or Markov decision process | Model Checking |

- HPN (High-level Petri Net).
- EH-CPN (Enhanced Hierarchical Color Petri Net).
- SXM (Stream X-machine Model).
- TLA (Temporal Logic Actions).
- EFSM (Extended Finite State Machine).
- LTS (Labeled Transition System).
- Flow graph-based test model.
- Markov chain diagram or Markov decision process.

Fig. 4 depicts the distribution of the model transformations used in the different testing approaches.[13] Fourteen (43%) of the testing approaches presented in the selected primary studies do not utilize any kind of model transformation. They rather rely on the source specifications when deriving test cases. In 4 (12%) of the testing approaches, the original specifications are transformed into Petri Net models. HPN and EH-CPN models (one test approach each) can be seen as derivatives of Petri Net models and consequently we can say that about 18% of the approaches use Petri Net models and its derivatives. This makes Petri Net models and their derivatives the mostly used models, followed by B models (6%), and SXM models (6%). All other models are equally used with 3% (one test approach) each.

Table 3 represents a matrix of the source specifications model and the transformation needed as well as the IDs of the primary studies where the transformation is used. A total of 15 testing approaches use an OWL-S model as a source model. No transformations are needed and the test cases are derived directly from the

[13] In Fig. 4 and in Table 3 we did not include the test approaches presented in PS9, PS19, PS21 and PS26 as they do not provide any test case derivation base specifications (Table 2).

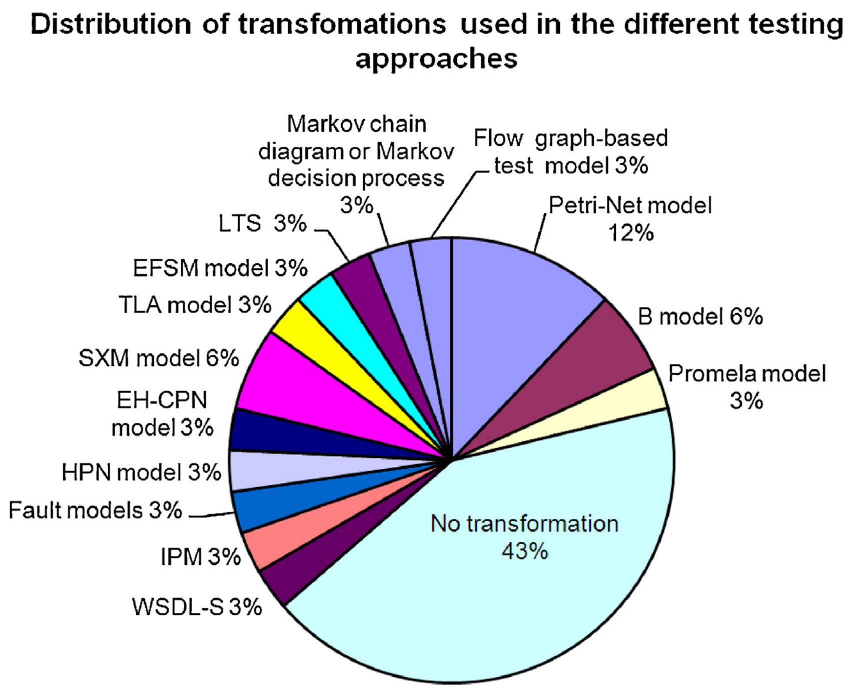


**Fig. 4.** Distribution of the model transformation used in the different testing approaches.

source specifications in test approaches where the source specifications model is an extended OWL-S model. In the selected primary studies, OWL-S models were extended to support mutant operators or they are augmented with a rule based specification model (SWRL) as in PS4 or FTLTL specifications as in PS24. In the test approach presented in PS5, the source specification is in WSDL, which provides only a syntactical description of the web service.

The WSDL representation is then enriched with the pre- and post- conditions for the service rule, which are specified using OCL. The enriched WSDL is then used to generate a semantic WSDL (WSDL-S) representation. Similarly, the source specifications for PS32 are described in terms of WSDL supported with additional IOPE information. In PS7 and PS30, the source specifications models are in WSDL-S augmented with a rule-based model (in SWRL and OCL respectively). These models are transformed to IPM and SXM models respectively. WSMO models are used as source specification models in PS2, PS15, and PS28. In PS2 and PS15, the source model is transformed into a B model. Studies PS2 and PS15 probably represent a continuation of the same work since both studies have same first author and almost share the same context. No model transformation was used in PS28.

#### 4.2.3. *Test case derivation techniques*

Different test case derivation techniques are used as a part of the test approaches introduced in the primary studies (Table A.2 in Appendix A and Table 2). Fig. 5 depicts the distribution of the test case derivation techniques. With 18%, model checking is the most popular technique (used in PS2, PS3, PS11, PS15, PS17 and PS34). As previously mentioned in Section 4.2.2, PS15 can be considered as the continuation of the work done in PS2. In both studies, model checking is used to generate test cases from a B model. In PS3, model checking is applied to a Promela model. In PS11 and PS17, model checking is applied to High-level Petri Net (HPN) model and Temporal Logic Actions (TLA) model respectively. Path traversing and reasoning over IOPE, random testing and mutation testing come in second place in terms of popularity with 9% (3 test approaches) for each. Path traversing and reasoning over IOPE is used in the primary studies PS1, PS12, and PS22, which all use a similar approach that first transforms OWL-S specifications into a Petri Net model and applies path traversing and reasoning over IOPE techniques to derive test cases. Random testing is used in PS6, PS14-1, and PS14-3 to derive test cases. In all three approaches, random testing is applied to the base specification model and no transformation is required. Mutation testing was utilized in PS10, PS20, and PS25 to derive test cases. In all of these studies, the base specification model is either OWL-S or an extension of it and there is no transformation into another specifications model. PS30 uses a test case derivation technique that is insufficiently described because of space limitations. Each of the remaining test case derivation techniques shown in Fig. 5 is used in only one primary study (3% each).

#### 4.2.4. *Test tool support*

Not all test approaches presented in the selected primary studies are supported by test tools. Around half of the testing approaches presented in the studies do not use any testing tools, as can be seen in Table 4. Six approaches (17.7%) in PS2, PS3, PS11 PS15, PS17, and PS34 use model checkers. Jess[14] (Java Expert System Shell) is used in PS4 to perform automated analysis over OWL and SWRL specifications. In PS5, the test tool WebInject[15] is used as a test runner and for report generation purposes. TAG-WS (Testing by Automatically Generated Web Service semantic) is used in PS6 as a prototypical implementation tool for the testing approach presented in the study. PS16 uses a converter to convert the identified test cases into JUnit[16] format to run it using that framework. In PS18, there are four different techniques presented for test case derivation from the EFSM model. The tool Gotcha (Friedman et al., 2002) is used only for the technique "User defined test objectives." PS19 uses the tool MWB[17] (Mobility Workbench) to detect requirements evolution. Therefore, we did not consider it as a test tool. AspectJ[18], used in PS20 and PS25, is not a testing tool but a tool that

[14] http://www.jessrules.com.
[15] http://www.webinject.org.
[16] http://www.junit.org.
[17] http://www.it.uu.se/research/group/mobility/mwb.
[18] http://www.eclipse.org/aspectj.

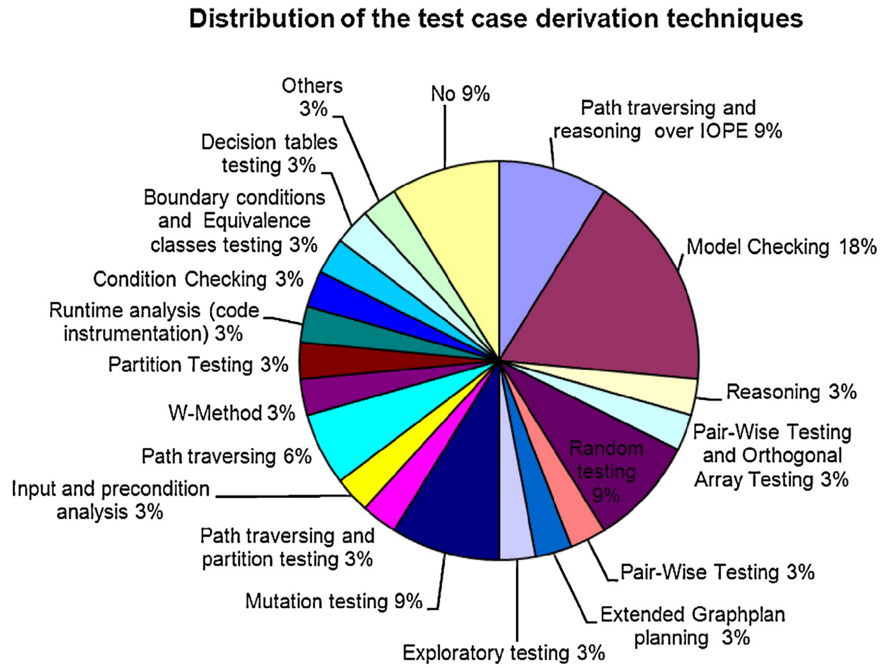


**Fig. 5.** Distribution of the test case derivation techniques.

supports the implementation of aspect orientation. Java-MOP[19] (Java Monitoring-Oriented Programming) is used in PS24 to support code instrumentation. In PS29 and PS30 the tools TAD (Testing by Automatically generate Decision table) and SWSDSXMGen are used respectively. Both tools are prototypically implemented to support the approaches presented in the respective papers.

#### 4.2.5. *Validation of the testing approach*

Most of the test approaches introduced in the selected primary studies use some kind of validation to provide confidence on the proposed approaches as shown in Table 4. Only the test approaches in primary studies PS1, PS20, PS26, and PS31 are not validated at all. Furthermore, the validation approach for PS3 and PS11 is not clearly specified. Other 19 primary studies are validated using (simple) examples mainly for the purpose of illustration of the testing approach introduced. Primary studies PS10, PS13, PS15, PS17, PS21, and PS25 use case studies to demonstrate the validity of the testing approach. PS8 validate its testing approach by applying it to a case study in an industrial environment. Hence it exhibits stronger validation than only using a simple example. We could not identify any credible validation done for the test approaches presented in PS14 and PS24. PS14 relies on the authors opinion to claim validity of the proposed approach. PS24 is about a testing system, but it did not give any details that may make the validation done credible.

### 4.3. *Test case derivation challenges (RQ2)*

Most of the studies that discuss challenges associated with testing focus on general challenges that are applicable to the problem of testing traditional web services as well as semantic web services. Table 5 summarizes the web services testing challenges introduced in the primary studies selected. As a consequence of source code invisibility (C1), structural testing techniques (white-box testing) cannot be applied to semantic web service. The absence of a (graphical) user interface (C2) prevents applying GUI-based testing approaches. This restricts noticeably the choices of the testers. Mapping from high level semantic description to low level syntactic description (C3) introduces overheads in the testing process. Traditional web services are described in WSDL that syntactically defines operations and massages structure. Conversely, semantic web services are described semantically using one of the ontology languages (e.g., WSMO and OWL-S). This semantical description needs to be "grounded", i.e., mapped to a technical description by defining message structure and operations of the web service in terms of WSDL. In the case of traditional web services, testing is carried out directly at the technical layer (i.e., through the interfaces described in WSDL). On the other hand, semantic web services are tested at the semantical layer (i.e., through the semantically described interfaces) using the available semantic information. This introduces more testing overhead than in the case of traditional web services (e.g., test pre- and post- conditions). Additionally, the poor observability and controllability (C4) is due to the dynamic and autonomic nature of the web services, which complicates the observation of the test results and the control of the testing process.

## 5. Discussion and implications for further research

Most of the identified test approaches in the primary studies present approaches for deriving functional test cases from the semantic web services requirement specifications. Some of these approaches (14) derive the test cases directly from the original specifications. Other 19 approaches require representing the original specifications in a different formalism using transformations before deriving the test cases. Many of them focus on different formalisms making their research efforts scattered in different directions. Focusing on one appropriate formalism might leverage the status of research in the area of functional testing of semantic web services. According to our review results, Petri Net is the most popular formalism, and therefore it is a good candidate for focusing on in future research. Although researchers can also investigate approaches that do not use an intermediate transformation, we do not recommend this direction. This is because semantic web services are specified using one of the semantic web services ontology languages (WSMO, OWL-S, WSDL-S, etc.) and most of these

[19] http://code.google.com/p/javamop/downloads/list.

**Table 3**
A matrix of test derivation base specifications and transformations used for test generation in the different primary studies. *Note*: Numbers refer to the numerical part of the primary studies IDs.

| Source specifications | Transformation into | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Petri Net model | B model | Promela model | WSDL-S | IPM | Fault models | HPN model | EH-CPN model | SXM model | TLA model | EFSM model | LTS | Flow graph-based test model | Markov chain diagram or Markov decision process | No transformation |
| OWL-S specifications | 1, 12, 22 | | 3 | | | 8 | 11 | 13 | | 17 | | | 31 | 34 | 10, 14-3, 20, 23, 33 |
| OWL-S and SWRL specifications | | | | | | | | | | | | | | | 4 |
| OWL-S, FTLTL and SWRL specifications | | | | | | | | | | | | | | | 24 |
| OWL-S specifications (extended with mutant operators support) | | | | | | | | | | | | | | | 25 |
| OWL and RIF-PRD specifications | | | | | | | | | 16 | | | | | | |
| WSDL and OCL specifications | | | | 5 | | | | | | | | | | | |
| WSDL specifications | | | | | | | | | | | | | | | 14-1 |
| WSDL and IOPE information | | | | | | | | | | | | | | | 32 |
| WSDL-S and SWRL specifications | | | | | | | | | 30 | | | | | | 6, 29 |
| WSDL-S and OCL specifications | | | | | 7 | | | | | | | | | | |
| WSDL-S specifications | | | | | | | | | | | 18 | | | | 14-2 |
| BPEL4WS specifications | 14-4 | | | | | | | | | | | 27 | | | |
| WSMO specifications | | 2, 15 | | | | | | | | | | | | | 28 |

**Table 4**
Validation approaches and tool support for the test approaches.

| Primary study ID | Tool support | Validation |
|---|---|---|
| PS1 | No | No |
| PS2 | ProB model checker | Example |
| PS3 | SPIN model checker | Not clearly specified |
| PS4 | Jess reasoning engine | Example |
| PS5 | WebInject | 4 Examples |
| PS6 | TAG-WS | Example (simple) |
| PS7 | No | Example (simple) |
| PS8 | No | Industrial application |
| PS9 | No | Example |
| PS10 | No | Case study |
| PS11 | HPN model checker (not clearly specified) | Not clearly specified |
| PS12 | No | Example |
| PS13 | No | Case study |
| PS14 | No | No credible validation |
| PS15 | Model checker (ProB) | Case study |
| PS16 | JUnit/Converter to JUnit | Example |
| PS17 | TLC model checker TLC | Case study |
| PS18 | Gotcha | Example |
| PS19 | No | Example |
| PS20 | AspectJ | No |
| PS21 | No | Case study |
| PS22 | No | Example |
| PS23 | No | Example |
| PS24 | Java-MOP | No credible validation |
| PS25 | AspectJ/FIT | Case study |
| PS26 | No | No |
| PS27 | No | Example |
| PS28 | No | Example |
| PS29 | TAD | Example |
| PS30 | SWSDSXMGen | Example |
| PS31 | No | No |
| PS32 | No | Example |
| PS33 | No | Example |
| PS34 | Model checker (PRISM) | Example |

specifications are semi-formal. Transforming them into formal models allows for comprehensive and automated testing.

Additionally, around 18 (50%) of the primary studies do not mention any test tool support. If the techniques involved in these studies do not actually use any test tools, there may be a significant reduction in the effectiveness of the approaches presented, since many tasks need to be done manually.

In what follows, we provide a comparison of two different test approaches found in the primary studies PS13 and PS14-4. Because of space limitations, we found it necessary to restrict the comparison to two test approaches. These approaches share the same characteristics, as they utilize transformations into Petri Net or its variant Enhanced Hierarchical Color Petri Net (EH-CPN) and the path traversing technique to identify the required test cases. PS13 introduces a test case derivation technique that involves transforming the OWL-S specification of a semantic web service into a EH-CPN and then analyzing control-flows and data-flows to identify all output-input-define-use chains. The chains are used to additionally identify corresponding executable paths in the EH-CPN model. Test sequences are derived from the executable paths in the

**Table 5**
Challenges with testing semantic web services.

| ID | Challenge/issue | Primary study |
|---|---|---|
| C1 | Source code invisibility | PS5,13 |
| C2 | No user interface | PS15 |
| C3 | Mapping from high level semantic description to low level syntactic description | PS15 |
| C4 | The poor observability and controllability | PS21 |

model. Test data are generated using the XML-based partition testing method in which XML structures and data types are mapped to category partitions and then test data are generated randomly based on the partitions. Test cases are generated by combining test sequences and test data. In PS14-4, a test data derivation approach for composite web services described in BPEL4WS is introduced. The approach involves converting BPEL4WS Specifications into a Petri Net model and applying path traversing to generate the test sequences. Test data are randomly generated based on the analysis of the data types available in the model.

It is clear that the test approaches presented in PS13 and PS14-4 use different transformations of their source specifications, namely, into EH-CPN and Petri Net respectively. Colored Peri Net (CPN) is a variant of Petri Net that utilizes colored tokens. Using color tokens allows for describing more complex data objects and remove part of the ambiguity in Petri Net. However, CPN is not capable of representing complex composition patterns of web services. Therefore, EH-CPN is introduced to solve this problem. As a result, EH-CPN is better than Petri Net as a target representation model for semantic web services that are composed of other web services (composite services). On the other hand, PS14-4 uses BPEL4WS to describe web services that are composed of other web services. The source specification in BPEL4WS is transformed into Petri Net which is less expressive than EH-CPN used in PS13 when it comes to composite web services. The test derivation technique followed in PS13 allows for directly deriving test sequences by traversing the execution paths in the EH-CPN model and generating the required test data by analyzing the input and outputs to identify category partitions and then generating test data based on them. In PS14-4, test sequences are identified by traversing the paths in Petri Net as in the approach used in PS13. However, the test data generation may result in relatively larger volume of test data compared to the test data generation approach used in PS13 which rely on the category partition technique.

As a result of the above discussion, we propose the following directions for future research in this area:

1 Focusing on Petri Net as a common formalism and developing rules for transforming OWL-S and WSMO specifications into their equivalent Petri Net representation.
2 Studying comparatively the different test approaches presented in the primary studies that share the same characteristics.
3 Investigating new approaches for self-adaptive testing where the test cases are automatically adapted whenever the requirements specifications of the semantic web services change.
4 Developing tools that support the research directions listed above (e.g., a tool for automating the transformation into Petri Nets).

## 6. Threats to validity

Validity is a main concern in empirical software engineering studies. Here, we discuss threats to construct, internal and external validities (Wohlin et al., 2000).

Construct validity is about whether the implementation of this systematic literature review matches its initial purpose. We identify the search process and search terms as the main concerns. The search terms used in this review were derived from the research questions and were tried against a list of known research studies and iteratively adjusted. However, the completeness and the comprehensiveness of the terms used are not guaranteed. To reduce this risk we searched the references list of each selected primary study to identify additional relevant primary studies. Additionally, the search revealed three articles in the Chinese language (Ju et al., 2008; Xiaoyan et al., 2008; Ying et al., 2009) which we excluded. This may present a threat to the construct validity. Although well-known digital libraries were searched for relevant primary studies, other sources may contain relevant primary studies that have not been taken into consideration.

Internal validity is the extent to which the design and conduct of the study are likely to prevent systematic error (Kitchenham and Charters, 2007). We are here concerned about the data extraction. When extracting data from the selected primary studies, we could only rely on our interpretation where the necessary data are not clearly expressed. Some required data were totally missing in a few primary studies. This may pose a threat to the internal validity.

External validity is the extent to which the effects observed in the study are applicable outside of the study (Kitchenham and Charters, 2007). External validity is primarily about the generalizability of the study outcomes. Our systematic review is constrained by the following issues: (1) it is focused on a very specific problem; (2) the problem under focus is relatively new, and (3) it covers a predefined period of time (2000–2011). Taking these concerns into account, we consider our results generalizable.

## 7. Conclusions

The systematic literature review reported in this work was carried out to acquire knowledge on the state of the art in the area of testing semantic web services and identify implications for future research. The focus was on finding answers to the research questions identified (Section 3.1). We could identify 34 relevant primary studies. The relatively small number of the identified primary studies can be explained by the fact that semantic web services emerged only around the end of year 2000 after the introduction of the first ontology language for the web (DAML+OIL)[20] (DAML, 2000). However, we believe that the identified primary studies present sufficient material to provide answers to the research questions under focus (Section 3.1).

Our results show that it is possible to derive test cases from requirements specification, based on the different testing approaches identified in the primary studies (RQ1). Around half of the test approaches analyzed start from an OWL-S specification model. For more than half of the approaches, transformation into another representation model was required. Base models are transformed to other representation models that are considered to be more efficient in terms of automatic test case generation or well supported by test case generation tools. In these approaches, test cases are derived from the transformed model. In the other approaches where transformation is not required, test cases are derived directly from the base model. The transformation used the most involves Petri Nets and its derivatives. In order to derive test cases, different techniques are applied to the specification models. Model checking is largely used to derive test cases.

As for RQ2, most of the studies that discuss challenges associated with testing semantic web services focus on general challenges that are applicable to the problem of testing traditional (syntactical) web services as well as semantic web services.

Testing semantic web services is a relatively new research area. We believe that much work remains to be done to improve the current state of research in the area of testing semantic web services following the future research directions proposed in Section 5.

## Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jss.2013.06.064.

[20] http://www.daml.org.

## References


Alderson, P., Green, S., Higgins, J. (Eds.), 2003. Cochrane Reviewers' Handbook 4.2.1. The Cochrane Collaboration, http://www.cebp.nl/media/m67.pdf (accessed April 2012).

Askarunisa, A., Abirami, A.M., MadhanMohan, S., 2010, July. A test case reduction method for semantic based web services. In: 2010 Second International conference on Computing, Communication and Networking Technologies, Karur, India, pp. 1–7.

Askarunisa, A., Punitha, K.A., Abirami, A.M., 2011, June. Black box test case prioritization techniques for semantic based composite web services using owl-s. In: 2011 International Conference on Recent Trends in Information Technology (ICRTIT), IEEE, pp. 1215–1220.

Bai, X., Kenett, R.S., 2009. Risk-based adaptive group testing of semantic web services. In: Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference, COMPSAC '09, vol. 2, IEEE Computer Society, Washington, DC, USA, pp. 485–490.

Bai, X., Lee, S., Tsai, W.-T., Chen, Y., 2008. Ontology-based test modeling and partition testing of web services. In: Proceedings of the 2008 IEEE International Conference on Web Services., IEEE Computer Society, Washington, DC, USA, pp. 465–472.

Bai, X., Wang, Y., Dai, G., Tsai, W.-T., Chen, Y., 2007. A framework for contract-based collaborative verification and validation of web services. In: Proceedings of the 10th International Conference on Component-based Software Engineering, CBSE'07., Springer-Verlag, Berlin, Heidelberg, pp. 258–273.

Bozkurt, M., Harman, M., Hassoun, Y., 2013. Testing and verification in service-oriented architecture: a survey. Software Testing, Verification and Reliability 23 (4), 261–313.

Chen, B., Zeng, G., Xu, H., 2009. The binary behavioral modes based on action sequence and compliance verification for compositional web service. In: Proceedings of the 2009 Asia-Pacific Conference on Information Processing, APCIP '09, vol. 01, IEEE Computer Society, Washington, DC, USA, pp. 415–419.

Dai, G., Bai, X., Wang, Y., Dai, F., 2007. Contract-based testing for web services. In: Proceedings of the 31st Annual International Computer Software and Applications Conference, COMPSAC '07, vol. 01, IEEE Computer Society, Washington, DC, USA, pp. 517–526.

DAML, 2000. Daml+oil. http://www.daml.org/2000/12/daml+oil-index.html (accessed April 2012).

Denaro, G., PezzA, M., Tosi, D., 2009. Ensuring interoperable service-oriented systems through engineered self-healing. In: Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering, ESEC/FSE '09, ACM, pp. 253–262.

Dong, W., 2009. Construction and test of web service solution for e-government. In: Proceedings of the 2009 International Conference on Computational Intelligence and Natural Computing, vol. 02., IEEE Computer Society, Washington, DC, USA, pp. 221–224.

Dyba, T., Dingsoyr, T., Hanssen, G.K., 2007, September. Applying systematic reviews to diverse study types: an experience report. In: First International Symposium on Empirical Software Engineering and Measurement, 2007, ESEM 2007, IEEE, pp. 225–234.

Engström, E., Runeson, P., 2011. Software product line testing—a systematic mapping study. Information and Software Technology 53 (1), 2–13.

Engström, E., Runeson, P., Skoglund, M., 2010. A systematic review on regression test selection techniques. Information and Software Technology 52 (1), 14–30.

Friedman, G., Hartman, A., Nagin, K., Shiran, T., 2002. Projected state machine coverage for software testing. In: ACM SIGSOFT Software Engineering Notes, vol. 27, ACM, pp. 134–143.

Goli, N.R., Pathari, V., 2006, December. A general framework for automatic verification of web services. In: 2006 International Conference on Advanced Computing and Communications, Mangalore, India, pp. 445–448.

Grigorova, V., 2006. Semantic description of web services and possibilities of bpel4ws. Information Theories and Applications 13 (2), 183–187.

Hull, J., Keane, J., Tell, L., Ernest, H., 2008. Development of 37 microsatellite loci for the great gray owl (strix nebulosa) and other strix spp. owls. Conservation Genetics 9 (5), 1357–1361.

IEEE, 1990. IEEE Standard Glossary of Software Engineering Terminology.

JabRef, 2011. Jabref bibliography reference manager. http://jabref.sourceforge.net/ (Accessed April 2012).

Jokhio, M.S., Dobbie, G., Sun, J., 2009. Towards specification based testing for semantic web services. In: Proceedings of the 2009 Australian Software Engineering Conference, ASWEC '09., IEEE Computer Society, Washington, DC, USA, pp. 54–63.

Ju, C., Di, W., Bixin, L., 2008. Extended hierarchical color petri net-based test case generation for composite services. Journal of Southeast University (Natural Science Edition) 4, 012.

Kitchenham, B., 2004. Procedures for Performing Systematic Reviews. Tech. Rep. Keele University.

Kitchenham, B., Charters, S., 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering. Tech. Rep. Keele University and Durham University Joint Report.

Lee, S., Bai, X., Chen, Y., 2008. Automatic mutation testing and simulation on owl-s specified web services. In: Proceedings of the 41st Annual Simulation Symposium (anss-41 2008)., IEEE Computer Society, Washington, DC, USA, pp. 149–156.

Li, B., Ji, S., Qui, D., Cai, J., 2010. Generating test cases of composite services based on owl-s and eh-cpn. International Journal of Software Engineering and Knowledge Engineering 20 (07), 921–941.

Liu, C., Chen, S., Kuo, J., Huang, T., 2011. A flow graph-based test model for owl-s web services. In: 2011 International Conference on Machine Learning and Cybernetics (ICMLC), vol. 2, IEEE, pp. 897–902.

Ma, C., Wu, J., Zhang, T., Hu, F., 2010. Web services testing based on stream x-machine. In: Proceedings of the 2010 10th International Conference on Quality Software, QSIC '10., IEEE Computer Society, Washington, DC, USA, pp. 232–239.

Mandell, D., McIlraith, S., 2003. Adapting BPEL4WS for the semantic web: the bottom-up approach to web service interoperation. In: The Semantic Web, ISWC 2003: Second International Semantic Web Conference Proceedings, vol. 2, Sanibel Island, FL, USA, October 20-23, 2003. Springer-Verlag New York Inc., p. 227.

Noikajana, S., Suwannasart, T., 2008. An approach for web service test case generation based on web service semantics. In: Proceedings of the International Conference on Semantic Web and Web Services, pp. 171–177.

Noikajana, S., Suwannasart, T., 2008b. Web service test case generation based on decision table (short paper). In: Proceedings of the 2008 The Eighth International Conference on Quality Software., IEEE Computer Society, Washington, DC, USA, pp. 321–326.

Noikajana, S., Suwannasart, T., 2009. An improved test case generation method for web service testing from WSDL-S and OCL with pair-wise testing technique. In: Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference, vol. 01., IEEE Computer Society, Washington, DC, USA, pp. 115–123.

OASIS, 2007. Web services business process execution language version 2.0. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel (Accessed April 2012).

Oghabi, G., Bentahar, J., Benharref, A., 2011. On the verification of behavioral and probabilistic web services using transformation. In: Proceedings of the 2011 IEEE International Conference on Web Services, ICWS '11., IEEE Computer Society, Washington, DC, USA, pp. 548–555.

Palacios, M., Garca-Fanjul, J., Tuya, J., 2011. Testing in service oriented architectures with dynamic binding: A mapping study. Information and Software Technology 53 (3), 171–189.

Paradkar, A.M., Sinha, A., Williams, C., Johnson, R.D., Outterson, S., Shriver, C., Liang, C., 2007. Automated functional conformance test generation for semantic web services. In: IEEE International Conference on Web Services, IEEE Computer Society, pp. 110–117.

Park, Y., Jung, W., Lee, B., Wu, C., 2009. Automatic discovery of web services based on dynamic black-box testing. In: Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference, vol. 01., IEEE Computer Society, Washington, DC, USA, pp. 107–114 http://portal.acm.org/citation.cfm?id=1632705.1633114

Pezzè, M., Young, M., 2007. Software Testing and Analysis: Process, Principles and Techniques. Wiley, ISBN 978-0471455936 http://www.amazon.com/Software-Testing-Analysis-Principles-Techniques/dp/0471455938#_

Ramollari, E., Kourtesis, D., Dranidis, D., Simons, A.J., 2009. Leveraging semantic web service descriptions for validation by automated functional testing. In: Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications, ESWC 2009 Heraklion., Springer-Verlag, Berlin, Heidelberg, pp. 593–607.

Rusli, H., Ibrahim, S., Puteh, M., 2011. Testing web services composition: a mapping study. Communications of the IBIMA 2011.

Shaban, M., Dobbie, G., Sun, J., 2009. A framework for testing semantic web services using model checking. In: Proceedings of the 2009 Fourth South-East European Workshop on Formal Methods, SEEFM '09., IEEE Computer Society, Washington, DC, USA, pp. 17–24.

Sinha, A., Paradkar, A., 2006. Model-based functional conformance testing of web services operating on persistent data. In: Proceedings of the 2006 Workshop on Testing, Analysis, and Verification of Web Services and Applications, TAV-WEB '06., ACM, New York, NY, USA, pp. 17–22.

W3C, 2004. OWL-S: semantic markup for web services. http://www.w3.org/Submission/OWL-S/ (Accessed April 2012).

W3C, 2004. Web services glossary. http://www.w3.org/TR/ws-gloss/defs (Accessed April 2012).

W3C, 2005. Web service semantics - WSDL-S. http://www.w3.org/Submission/2005/SUBM-WSDL-S-20051107/ (Accessed April 2012).

Wang, H., Li, L., Wang, C., Kang, Z., Liu, D., Wu, J., Bouguettaya, A., 2009, December. Logic-based verification for web services composition with tla. In: 2009 IEEE International Conference on Service-Oriented Computing and Applications (SOCA), Taipei, Taiwan, pp. 1–8.

Wang, R., Huang, N., 2008. Requirement model-based mutation testing for Web service. In: Next Generation Web Services Practices, 2008. NWESP'08. 4th International Conference on. IEEE, pp. 71–76.

Wang, H.h., Sun, J., 2009, December. A semantic web environment for components. The Knowledge Engineering Review 24, 59–75.

Wang, X., Huang, N., Wang, R., 2009b. Mutation test based on owl-s requirement model. In: Proceedings of the 2009 IEEE International Conference on Web Services, ICWS '09., IEEE Computer Society, Washington, DC, USA, pp. 1006–1007.

Wang, Y., Bai, X., Li, J., Huang, R., 2007. Ontology-based test case generation for testing web services. In: Proceedings of the Eighth International Symposium on Autonomous Decentralized Systems, IEEE Computer Society, Washington, DC, USA, pp. 43–50.

Wen-Jie, Y., Shi, Y., 2009, December. Modeling requirements evolution with $\pi$-calculus. In: 2009 2nd International Conference on Power Electronics and Intelligent Transportation System (PEITS), vol. 3, IEEE, pp. 355–358.
Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A., 2000. Experimentation in Software Engineering: An Introduction. Kluwer Academic Publishers, ISBN 978-1-4613-7091-8, http://dx.doi.org/10.1007/978-1-4615-4625-2 http://link.springer.com/book/10.1007/978-1-4615-4625-2/page/1
WSMO, 2004. Web service modeling ontology. http://www.wsmo.org/ (Accessed April 2012).
Xiaoyan, Z., Ning, H., Ying, Y., 2008. Owl-s based test case generation. Journal of Beijing University of Aeronautics and Astronautics 3, 019.
Ying, Y., Maozhong, J., Ning, H., 2009. Testing control flow of composite service. Journal of Beijing University of Aeronautics and Astronautics 35, 117–121.
Yu, B., Li, D., 2010 Oct. Generating test data based on xml schema. In: 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), Taiyuan, China, pp. 344–348.
Yu, Y., Huang, N., Luo, Q., 2007. OWL-S based interaction testing of web service-based system. In: Proceedings of the Third International Conference on Next Generation Web Services Practices, NWESP '07, IEEE Computer Society, Washington, DC, USA, pp. 31–34.
Zakaria, Z., Atan, R., Ghani, A., Sani, N., 2009. Unit testing approaches for BPEL: a systematic review. In: Proceedings of the 2009 16th Asia-Pacific Software Engineering Conference, IEEE Computer Society, pp. 316–322.
Zhang, T., Yao, Q., Zheng, X., Ma, C., Wang, H., 2011. An approach of end user regression testing for semantic web services. In: Management and Service Science (MASS), 2011 International Conference on, IEEE, pp. 1–4.
Zhang, Y., Zhu, H., 2008. Ontology for service oriented testing of web services. In: Proceedings of the 2008 IEEE International Symposium on Service-Oriented System Engineering., IEEE Computer Society, Washington, DC, USA, pp. 129–134.

**Abbas Tahir** received his masters degree in Information and Communication Systems from the Hamburg University of Technology, Hamburg, Germany. He is currently working towards the PhD degree at the Department of Theoretical and Applied Sciences, University of Insubria, Como, Italy. He has many years of industrial and research experience in the field of Software Testing and Quality Engineering, working for Siemens AG, Corporate Technology, Munich, Germany. Abbas Tahir has been involved in several international research projects such as ASG, QualiPSo and SPaCIoS. His research interests include software verification and validation, quality assurance and model-based testing.

**Davide Tosi** received his MSc in Computer Science (2002) and his PhD in Computer Science (2007) at the University of Milano Bicocca. Currently Davide Tosi is a research fellow at the Università degli Studi dell'Insubria and charter member of OpenSoftEngineering. His research interests include software testing, mobile agens, self-managed software, and open source quality. He has published about 30 research papers in international journals, conferences and workshops. Davide Tosi has been involved in several international projects, including EU-funded projects such as SHADOWS, QualiPSo, and SUPERHUB. Davide Tosi has served on the PC of a number of international software conferences.

**Sandro Morasca** is a professor of Computer Science at the Università degli Studi dell'Insubria in Como and Varese, Italy. He has been actively carrying out research in Empirical Software Engineering, Specification of Concurrent and Real-time Software Systems, Software Verification, Open Source Software. He has published over 20 journal and 70 conference papers. Sandro Morasca has been involved in many projects, the latest of which was QualiPSo, the largest EU-financed project in the 6FP. He has served on the PC of many international Software Engineering conferences and serves on the Editorial Board of "Empirical Software Engineering: An International Journal" (Springer-Verlag).