This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

Model Checking Based Web Service Verification: A Systematic Literature Review

Gopal N. Rai[1,2] and G. R. Gangadharan[1] *Senior Member, IEEE*
[1]IDRBT, Hyderabad, India
[2]University of Hyderabad, Hyderabad, India

Model checking is a popular formal technique facilitating automatic verification of finite-state transition systems, and it has been applied to almost all of the Web service verification aspects, such as control-flow, data-flow, interaction, time requirements, quality of service, security requirements, etc. However, no systematic literature review focused on model checking of Web services is available. Motivated by this fact, in this paper, we systematically review existing research works on model checking based Web service verification appeared during the period of 2002 – 2017. We present a verification goal based classification of the collected articles, and for each paper, we identify the verification technique, target Web service standard, flavor of the employed model checking technique and tool, and other supplementary techniques, if any. Further, we highlight some of the issues, gaps, key challenges in this area and propose some future directions.

***Index Terms*—Web service, verification, model checking, systematic literature review**

I. Introduction

Model checking [1] is a popular formal technique facilitating automatic verification of finite-state transition systems, and it has been applied to almost all of the Web services verification aspects, such as control-flow, data-flow, interaction, time requirements, quality of service, security requirements, etc. Model checking verifies the desired behavioral properties of a finite-state transition system for a given model through complete exploration of all the reachable states and the properties that disseminate through them. In model checking, temporal logics (for instance, Linear Temporal Logic (LTL) and Computation Tree Logic (CTL)) are used for specifying the properties of a finite-state transition system. A temporal logic uses atomic propositions and Boolean connectives to build up complicated expressions describing the properties of a reactive system. Developed as per the requirements, various flavors of model checking with their accompanying tools are available, such as bounded model checking, probabilistic model checking, abstract model checking, timed model checking, etc.

A systematic literature review (or simply a systematic review, structured literature review or SLR) is a literature review focused on a research topic that tries to identify, classify, select, and synthesize all high-quality research evidences relevant to that topic area [2]. An SLR is a structured process that identifies, evaluates, and interprets the existing evidences of research that are pertinent to a specific research query. Hence, it follows a predefined search category. An SLR provides a framework for locating new research tasks, discovers gaps in ongoing research, and recommends areas for further examination [2], [3]. As is witnessed by the literature, model checking is widely used in the area of Web service verification. However, so far, to the extent of our knowledge, no systematic literature review has been done on model checking based Web service verification. Unavailability of such SLR makes it cumbersome to evaluate the current research state and find the research gap in the area of Web services model checking. Though, in the literature, we find a number of survey papers on Web service verification (for instance, [4], [5], [6], [7]) that discuss model checking too, a separate discussion focused on Web services model checking is required.

Corresponding author: G.R.Gangadharan (email:geeyaar@gmail.com)

Motivated by the said facts, the objective of this paper is to provide a state-of-the-art review of model checking based Web service verification proposals that explore: (i) main research motivations behind Web service model checking, (ii) the existing methods and techniques that support Web service verification and their verification goals, and (iii) existing research issues and future research directions in the area of Web service verification.

Consequently, this paper develops a verification goal based classification of the collected articles, and for each paper, it identifies the verification technique, target Web service standards, flavor of the employed model checking technique and tool, and other supplementary techniques, if any. Further, it highlights some of the issues, gaps, key challenges in this area and provides some future directions.

The rest of the article is organized as follows: our review methodology is explained in Section II. Section III presents the classifications and review of the verification approaches resulted from the methodical search. Section IV presents the results and analysis of our Systematic Literature Review (SLR). Research implications and future research directions are given in Section V, followed by the concluding remarks in Section VI.

II. Review Methodology

We follow the five-step process of conducting systematic review as described in [8] (see Fig. 1). The process is similar to the methodologies followed in [9], [2], and [3].

(1) Framing Research Questions. In order to conduct the present review, we frame the research questions and provide their motivations as listed in Table I.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

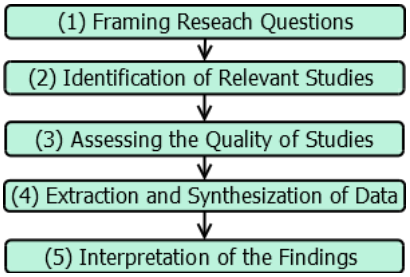

Fig. 1. Five-steps Process of Conducting Systematic Review (Based on [8])

Research Questions	Motivations
RQ1—Why is model checking a popular verification technique for Web services?	To get insight on model checking based Web service verification techniques satisfying functional and non-functional requirements.
RQ2—Which aspects are verifiable using model checking in the context of Web services?	To identify the properties of Web services that are being verified using model checking based techniques.
RQ3—What are the existing approaches, tools, and techniques that help and promote model checking based Web service verification?	For the identification, classification, and comparison of the existing approaches and techniques that are employed in model checking based Web service verification.
RQ4—What are the existing challenges, current research state, and the possible future research directions in the area of model checking based Web service verification?	To identify and explore the potential research challenges that have to be addressed, and to find the possible future research directions in this area.

TABLE I
RESEARCH QUESTIONS AND THEIR MOTIVATIONS

(2) Identification of Relevant Studies. Primarily, we referred articles from the popular online data sources, namely, including IEEE-Xplore, ACM digital library, Springer, Science Direct, Wiley, IGI Global, Inderscience, and Google Scholar. Non-peer-reviewed articles (such as white papers), editorials, abstracts, short papers (less than 4 pages), and non-English scripts are excluded from the scope of this survey. We also did not consider a book or book chapter or dissertation or extension paper or tool paper if the technique and results presented in the document are already published in a journal or conference. Also, semantic Web services are not considered for this study as it is a broad topic and requires a separate treatment.

For our search, we provided the search string as follows:

(Web service OR Web services OR service OR BPEL OR BPEL4WS OR choreography OR choreographies OR WSDL OR WS-CDL OR WSFL OR orchestration)

AND

(verification OR verifying OR validation OR checking OR model checking OR UPPAAL OR SPIN OR NuSMV)

(3) Assessing the quality of studies. We refined our search using various words like "model checking", "temporal logic", etc. We extracted nearly 582 papers for this survey that were published during 2002-2017 in various conferences and journals. The year 2002 was chosen as we did not find a paper on Web service verification published before 2002.

Initial selection. Our initial selection of 582 articles covered the research topic of Web service verification across the search databases. We explored the title, abstract, and keywords of the primarily selected studies and applied above-mentioned criteria of inclusion/exclusion.

Final selection. For our final selection, we focused particularly on model checking based Web service verification approaches among the articles collected in initial selection phase and finalized 105 studies for this review.

Fourth step of the systematic review *extraction and synthesization of data* is presented in Section III, and fifth step *interpretation of findings* is presented in Section IV and in Section V.

III. CLASSIFICATION OF THE MODEL CHECKING BASED WEB SERVICE VERIFICATION APPROACHES

There are various formal techniques available for the verification of Web services, such as model checking, Petri net, process algebra, etc. However, in this present review, we did not aim at presenting a systematic literature review on verification of Web services that includes all available verification techniques as it would be too broad to cover completely in a single paper. In this work, we focused only on the model checking based Web service verification approaches as model checking is a verification technique that is applied to almost all of the verification aspects related to Web services.

We classify the model checking based Web service verification approaches based on their verification goals classified into six classes: composition verification, behavioral verification, interaction verification, equivalence verification, non-functional requirement verification, and time requirement verification. Further, apart from these classes, there exist research papers on model checking based Web service verification such that they focus on multiple verification goals, and there exist research papers such that they do not belong to any of the said classes. Fig. 2 presents a classification of the existing Web service model checking approaches.

A. Web Services Composition Verification

In the flow model of a service composition, a composition is created by describing how to use the functionality provided by the collection of composed Web Services [10]. This is also known as flow composition, orchestration, or choreography of Web Services. For instance, verifying the existence of a deadlock in a composition scenario is an example of composition verification. Following example (based on [11]) presents a scenario where composition verification is required. A merchant is operating three hosts (e.g., for redundancy or load-balancing reasons). Two of these hosts ('Host 1' and 'Host 2') are used to run a web server, e.g., Apache. The web server itself is split up into multiple processes, $T1, \cdots, T4$. The Web server processes have joint access to a shared database, e.g., using the SQL protocol. This database is assumed to be the only form of communication between the server processes. The server processes may contact a third party, e.g., to authorize a payment. Incoming client requests are modeled by means of processes $T5$ and $T6$. Now, model

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

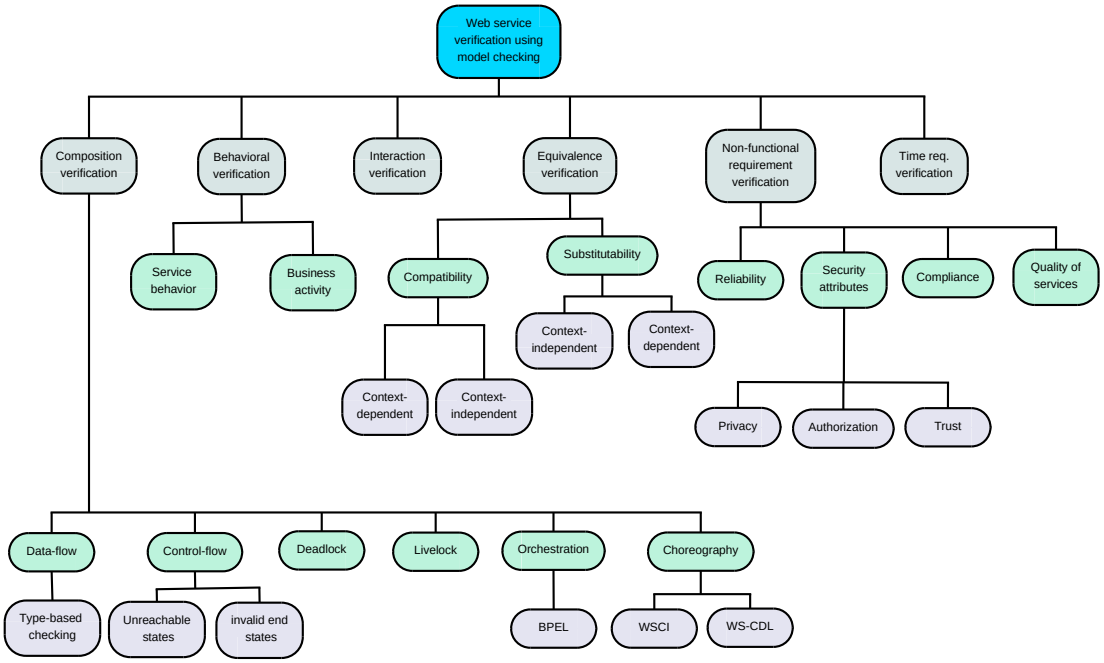

Fig. 2. Classification of the Approaches Based on Their Verification Goal

checking can be used to verify that the code does not get into a state in which it deadlocks or livelocks.

Dai et al. [12] annotated *Business Process Execution Language* (BPEL) and transformed it into Time Constraints Petri Net (TCPN) for the verification of control-flow aspects. Zhang et al. [13] classified the feature interaction into five classes: deadlock, livelock, invocation error, race condition, and resource contention. Corradini et al. [14] presented an approach for the verification of Web service compositions expressed in the *Web Services Choreography Description Language* (WS-CDL) language. Their technique analyzed the given WS-CDL source document and extracted the necessary information. Further, the extracted information was translated to PROMELA code and was verified against the given specification. Qian et al. [15] proposed an automated way to extract formal models from programs implementing Web services using predicate abstraction for abstract model checking.

Dong et al. [16] presented a formalized analysis model that accepts specification and semantics of business flow as inputs and provides asynchronous communication model as an output. Further, this output model is verified by using the Simple Promela INterpreter (SPIN) model checker. Luo et al. [17] proposed a model for BPEL based on its grammar and control flow, then they transformed the required part of BPEL into the input language of Model Checking Time and Knowledge (MCTK) and verified the specification properties. Barker et al. [18] presented a Multi-Agent Protocol (MAP) based Web service choreography language for specifying and verifying service interaction patterns. The advantage with their approach is that services do not have to be pre-configured at design-time. For the verification purpose, they proposed translation from MAP to PROMELA. Xu et al. [19] presented a technique for the verification of Web service choreography written in WS-CDL by mapping it to *Communicating Sequential Processes* (CSP) and using Process Analysis Toolkit (PAT) model checker. Luo et al. [20] developed a translation procedure to translate BPEL into I/OLTS, and then developed another translation procedure to translate I/OLTS into the input language of ZING model checker. Mei et al. [21] translated BPEL4WS into interface automata, which was then translated into PROMELA. Xinlin [22] proposed a Web service verification approach based on the category theory.

Zhang et al. [23] proposed a *Timed Asynchronous Communication Model* (TACM), which can precisely describe the timed properties and asynchronous communication behaviors. The advantage with TACM is that it can be directly provided as input to UPPAAL model checker. Li et al. [24] proposed a model checking based technique to automatically verify the composite services. For the property specification, they used universal Model Sequence Diagrams (uMSDs). Fu and Chen [25] addressed the problem of state explosion by introducing predicate abstraction and refinement technology in the traditional model checking method. Further, they proposed a framework for Web service composition based on the technology of abstraction and refinement. Luo et al. [26] proposed a technique for Web service verification that translates BPEL to BSTS and then BSTS to *Interpreted Systems Programming Language* (IPSL) which is input language for the model checker MCMAS. The advantage with their approach is that the epistemic and cooperation properties also can be specified and verified in their framework. Todica et al. [27] proposed a SPIN model checker based technique for verifying automatically generated business process. Zhao et al. [28] proposed a model for guaranteeing correctness of WS-BPEL. The input language for their model is Language Of Temporal Ordering Specification (LOTOS) and they check validity of the model by using model checking. Sharygina and Kröning [11] presented an automatic abstraction refinement based model checking

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

technique for Web services. Their technique formalized the

Studies	Verification Goal	WS Technology	Model Checker
Dai et al. [12]	Reachability, safety, liveness, and correctness	BPEL	N/A
Zhang et al. [13]	Detecting feature interactions	BPEL4WS	SPIN
Corradini et al. [14]	WS-CDL choreographies	WS-CDL	SPIN
Qian et al. [15]	Applications that implements services	N/A	N/A
Dong et al. [16]	Behavioral specification of the workflow	BPEL	SPIN
Luo et al. [17]	BPEL control flow	BPEL	MCTK
Barker et al. [18]	Termination of MAP service choreography	BPEL	SPIN
Xu et al. [19]	WS-CDL based specifications	WS-CDL	PAT
Luo et al. [20]	Correctness of Web services	BPEL	ZING
Mei et al. [29]	Correctness of BPEL4WS	BPEL	SPIN
Xinlin [22]	Control structure of service composition	N/A	N/A
Zhang et al. [23]	Asynchronous communication behavior and timed properties	N/A	UPPAAL
Li et al. [24]	Correctness of composite services	BPEL	SPIN
Fu and Chen [25]	Web service composition	N/A	SPIN
Luo et al. [26]	Web service composition	BPEL	MCMAS
Todica et al. [27]	Business Processes	BPEL	SPIN
Zhao et al. [30]	BPEL-based service composition	BPEL	CADP
Zahoor [31]	Declarative service composition processes	N/A	N/A
Sharygina and Kröning [11]	Concurrency, deadlock, and livelock	N/A	N/A
Yu et al. [32]	Concurrency in BPEL	BPEL	N/A

TABLE II
SUMMARY OF DIFFERENT STUDIES ON WEB SERVICE COMPOSITION VERIFICATION USING MODEL CHECKING

semantics of a PHP-like scripting language for implementing Web services by the means of labeled Kripke structures. However, it lacks the support for the verification of liveness properties. Yu et al. [32] presented a modular verification method for composite Web services which were written in BPEL. Their focus was on the interleaving semantics of BPEL and concurrently running processes. They used the symbolic encoding for modeling the processes. Table II presents a summary of different studies on Web service composition verification using model checking.

B. Web Services Behavioral Verification

Behavioral specification of a service composition expresses a causal relationship among various invocations by using control and data flow links [33]. Figure 3 presents a scenario where behavioral verification is required. In Figure 3, the flow

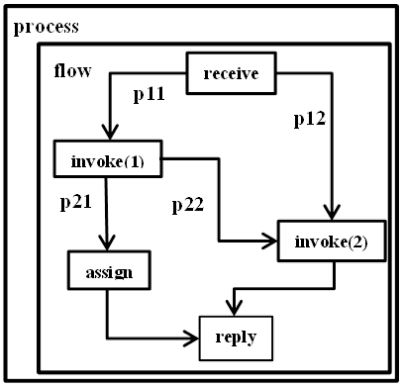

Fig. 3. An Example Scenario for Behavioral Verification (Based on [33])

has five atomic activities (receive, invoke(1), invoke(2), assign, and reply), that are executed concurrently, but have causal dependencies. Figure 3 uses variables such as $p11$ and $p12$ to denote the condition schematically. Once the execution control is passed to the flow activity, its inside activities start their execution concurrently. The example in Figure 3 requires dead path elimination for executing correctly. When both $p12$ and $p22$ turn out to be false, the join condition of the invoke(2) activity becomes false and the activity is never executed. Then the join condition of the reply activity (actually a logical or ($\vee$) of the two incoming links) is not evaluated because one of them comes from the invoke(2) activity. On the other hand, since the outgoing link from the assign activity becomes true, the reply activity will logically be executable.

Díaz et al. [34] presented the unification of WS-CDL and BPEL documents. Their aim was to generate the correct BPEL skeleton documents from the WS-CDL documents by using the timed automata as an intermediary model in order to check the correctness of the generated Web Services with the model checker UPPAAL. Yeung [35] presented mappings of WS-CDL and BPEL into CSP for specifying and verifying the behavior of Web services based business processes. Verification process can be carried out based on the notion of trace-refinement of CSP and can take the advantage of FDR2 model checker. Nakajima [33] proposed an approach to extract the behavioral specification from a BPEL application program and to analyze it by using the SPIN model checker. This approach considers Dead-Path Elimination (DPE) and the abstraction of control variables. Hallé [36] proposed a method for generating Web service stubs by using the first-order temporal logic, called LTL-FO+. In this approach, the model checker NuSMV is used as a back-end engine to produce message traces compliant with the requirement formulae specified in LTL-FO+. Bentakouk et al. [37] proposed a formal testing framework for addressing the issue of behavioral verification of service composition at design time. The proposed approach is based on the symbolic testing and an SMT solver.

Gao et al. [38] used Live Sequence Chart specifications (LSC) to specify the complex behaviors among multiple Web services, and then translated LSC to Extended Labeled Transition System (ELTS). They proposed a projection approach to check the behavioral correctness and consistency of functional

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

requirements against the ELTS model using NuSMV. Díaz et al. [39] presented a timed automata based generic model for publishing and managing Web service resources that includes operations for clients to discover and subscribe to resources, with the intention of being notified when the resource property values fulfill certain conditions. Further, they provided an error handling mechanism to deal with discovery failure and resource life expiration, and used UPPAAL to verify the model soundness. Table III presents a summary of different studies on Web service behavior verification using model checking.

Studies	Verification Goal	WS Technology	Model Checker
Díaz et al. [34]	Correct generation of BPEL	WS-CDL	UPPAAL
Yeung [35]	Behavioral specification	WS-CDL, BPEL	FDR2
Nakajima [33]	Behavioral specification	BPEL	N/A
Hallé [36]	Service behavior	N/A	NuSMV
Bentakouk et al. [37]	Behavioral conformance	BPEL	N/A
Gao et al. [38]	Behavioral correctness	N/A	NuSMV
Azaiez and Sbaï [40]	Behavioral properties	WS-CDL	NuSMV
Díaz et al. [39]	Publish-subscribe architecture	WSRF	UPPAL
Ravn et al. [41]	Business activity	N/A	UPPAAL
Marques et al. [42]	Business activity protocol	N/A	UPPAAL

TABLE III
SUMMARY OF DIFFERENT STUDIES ON WEB SERVICE BEHAVIOR VERIFICATION USING MODEL CHECKING

C. Web Services Interaction Verification

In a Web service composition scenario, an interaction pattern is a description of the overall partner interactions. The interactions are modeled as links between endpoints of the Web services' interfaces, each link corresponding to the interaction of one Web service with an operation of another Web service's interface. Let us consider a requirement specification formula ϕ (written as Eqn. 1), in which all the literals are Web service messages.

$$\phi = A((Flight\_Yes \wedge Flight\_Book) \rightarrow (Flight\_Booked \vee (\neg Hotel\_Booked \cup (Flight\_Booked)))) \quad (1)$$

The said requirement specification formula ϕ states that in all the traces, if flight is available and booking is requested, then either flight must be booked or hotel must not be booked until flight is booked.

Application of model checking based techniques for the verification of Web service interaction is bit old, but still is in use because of its efficacy. Foster et al. [43] proposed a model-based technique to verify Web service compositions represented in the form of BPEL. They modeled specifications in the form of Message Sequence Charts (MSCs). Further, BPEL and MSCs were mechanically compiled into the Finite State Process notation (FSP). Then, verification process takes place between FSPs generated from BPEL and MSCs using trace equivalence. Contrary to [43], Fu et al. [44] presented a Web service interaction verification scheme based on the centralized theme of conversation modeling. They specified the desired conversations of a Web service as a guarded automaton. Their focus was on the asynchronous messaging and they made effort to relax the restrictions in the way of direct application of model checking.

Kazhamiakin et al. [45] presented a specification and verification technique for Web services. They used formal tropos for specification modeling and PROMELA for process modeling. Walton [46] verified the interaction among agents participating in multi-agent Web service systems by proposing a Web service architecture and a lightweight protocol language. Further, he verified the specification properties written in the proposed language using model checking. Fu et al. [47] presented a realizability analysis framework that consists of three steps: (i) specification of a service composition using a realizable conversation protocol, (ii) verification of the desired LTL properties on the conversation protocol, and (iii) projection of the conversation protocol to each pair. Pistore et al. [48] presented a requirements-driven design and verification approach for Web services. They used a model checking based technique for validating specifications and verifying requirements. Techniques presented in [43], [44], [46] were efficient, however, they did not deal with the automation of verification process.

Cao et al. [49] presented a model-based verification framework for BPEL. BPEL is modeled by using UML activity diagram and verified using software model checking technique. Díaz et al. [50] presented an analysis technique for the timed behavior of Web services. Their technique is based on the translation of service description into the timed automata, and then, they used UPPAAL model checker to simulate and analyze the system behavior. Zheng et al. [51] presented a test case generation framework for BPEL using SPIN and NuSMV. They modeled BPEL as Web Service Automata (WSA) and on the basis of WSA, they presented their test case generation framework. Test cases were used to verify whether the implementation of Web services meet the pre-specified BPEL behavior. Deutsch et al. [52] presented a technique to verify the sequence of events (input, states, and action) resulting from the interaction of Web services in data-driven web applications.

Gu et al. [53] presented a method for verification of Web service conversations written in WSCL by using SPIN. Parizek and Adamek [54] presented a verification technique for checking the compliance of a composite service with the constraints on the order of operation invocations on stateful basic services. Shegalov and Weikum [55] described the generic recovery protocols in the interaction contracts framework that guarantees exactly-once execution and applied model checking to prove its correctness. Wan et al. [56] presented a multi-agent based verification technique for Web services, where Web service communities are modeled as UML activity diagrams. Specifications are expressed in CTL* and MCMAS model checker is used for verification. Qian and Chen [57] presented a model checking based testing technique for Web service composition. They used guarded automata for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

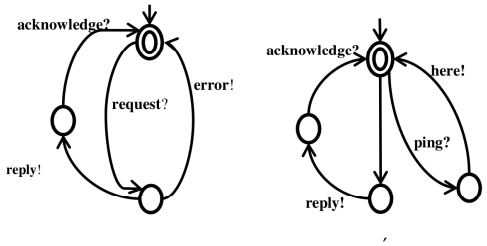

Fig. 4. An example of Equivalent/Substitutable Services (Based on [64])

specifying the service composition (written in WS-CDL) and SPIN for verifying the interactions of services.

Further, as an improvement over the previous ones, recent model-checking based verification techniques [58], [59], [60] support automation to a great extent. Bentahar et al. [60] proposed a modeling and verification technique for composite Web services. Their modeling aspect was based on separation of concerns between operational and control behaviors (interactions among Web services) of Web services. Their verification technique was model checking-based where they automatically generated Kripke model out of the given operational behavior. Similarly, Sheng et al. [59] proposed an automated service verification approach based on the operational and control behavior. The coordination of operational and control behaviors at run-time was facilitated by conversational messages and their proposed automated verification technique was based on symbolic model checking. El Kholy et al. [58] presented a framework to capture and verify the interactions among multi-agent based Web services. In order to capture the interactions, they proposed a specification language that uses commitment modalities in the form of contractual obligations. Table IV presents a summary of different studies on Web services interaction verification using model checking.

Studies	Verification Goal	WS Technology	Model Checker
Foster et al. [43]	Workflow trace equivalence verification	BPEL	N/A
Fu et al. [44]	Interaction among Web services	BPEL	SPIN
Kazhamiakin et al. [45]	Interaction among Web services	BPEL	SPIN
Walton et al. [46]	Mult-agent Web services	N/A	SPIN
Fu et al. [47]	Realizability of conversation protocol	N/A	N/A
Pistore et al. [48]	Verification of business process model	BPEL	NuSMV
Cao et al. [49]	Verification of UML model for BPEL	BPEL	SPIN
Deutsch et al. [61]	Communicating data-driven Web services	N/A	N/A
Zheng et al. [51]	BPEL test cases	BPEL, WSDL	SPIN, NuSMV
Deutsch et al. [52]	Data-driven Web applications	WDL	N/A
Gu et al. [53]	Web service conversations in WSCL	WSCL	N/A
Parizek and Adamek [54]	Session-oriented interactions	BPEL	Java PathFinder
Shegalov and Weikum [55]	Correctness of recovery framework	N/A	N/A
Wan et al. [56]	Web service communities	N/A	MCMAS
Qian and Chen [57]	Test case specifications	WS-CDL	SPIN
Bentahar et al. [60]	Web service operational behavior	N/A	N/A
Sheng et al. [59]	Web service operational behavior	N/A	NuSMV
El Kholy et al. [58]	Mult-agent based Web service compositions	N/A	MCMAS
El Kholy et al. [62]	Multi-agent based Web service compositions	N/A	MCMAS
Ghannoudi and Chainbi [63]	Interaction among Web services	WSDL	UPPAL

TABLE IV
SUMMARY OF DIFFERENT STUDIES ON WEB SERVICE INTERACTION VERIFICATION USING MODEL CHECKING

D. Web Services Compliance and Equivalence Verification

Dynamic reconfiguration of services in a composition scenario leads to the quest of whether available services can substitute an unavailable service. For example, Figure 2 depicts two equivalent services A and A'. Services are represented by a set S of states (the circles), transitions between the states (the arrows) and actions (the labels of the arrows) which can either be emissions or receptions. Sending a message n is written n! and receiving it is written n?, and we denote A as the set of actions of the form n! or n? (where n ranges over a predefined set of message names). The problem of service substitution is categorized as context-independent and context-dependent [64], [65]. Finding a context-independent substitute for a service is easy as it is computed directly on the basis of the WSDL documentations of services [66], whereas process of finding a context-dependent substitute is complicated and requires a more sophisticated procedure [67]. Since substitutability is not a stand-alone expression, it is always studied with other accompanying terminologies such as compatibility [68], [64], behavioral equivalence [69], [70], and compliance/conformance [71].

Pathak et al. [65] presented the concept of environment based substitutability notions: environment dependent and environment independent. Their logical formulation of the solution is based on the well-studied notion of "quotienting". Both and Zimmermann [72] analyzed Web service composability for compatibility, replaceability, and process conformance. Their approach enables one to verify if a newly added service holds the same rule as the previously existing one. For this purpose, they used the Process Algebra Nets (PANs). Salah et al. [73] proposed a technique to check whether a concrete executable BPEL process conforms to its published interface behavior. In the process, they translated the concrete BPEL processes into PROMELA model and interface expressions into trace assertions. Guermouche and Godart [74] proposed a model checking based framework to deal with verifying the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

compatibility of a choreography where asynchronous communicating Web services are considered. Bersani et al. [75] proposed a SMT-based formal technique for runtime checking of Web service substitutability. Yeung [76] proposed a notion of choreography conformance in terms of Communicating Sequential Processes. Further, by using the proposed notion, he proposed an model checking based approach for service composition and verification.

Camera et al. [77] proposed an interactive tool-based approach for specification and verification of Web services behavioral adaptation contracts. Groefsema et al. [78] presented an approach for preventive compliance checking using temporal logics and model checking techniques. In order to get the interpretation model for verification, they proposed a mapping of service compositions to Kripke structures by using colored Petri nets. Huynh et al. [79] proposed a logic-based technique for finding similarity between Web services. Bataineh et al. [80] proposed a model checking based technique for verifying the compliance of autonomous and intelligent agent-based services with contracts regulating their composition specified in BPEL. Table V presents a summary of different studies on Web services compliance and equivalence verification using model checking.

Studies	Verification Goal	WS Technology	Model Checker
Pathak et al. [65]	Context specific substitutability of services	N/A	N/A
Both and Zimmermann [72]	Recursive and parallel BPEL systems	BPEL, WSDL	N/A
Salah et al. [73]	Conformance between behavioral interface and WS-BPEL	WS-BPEL	SPIN
Guermouche and Godart [74]	Choreography compatibility analysis	N/A	UPPAAL
Bersani et al. [75]	Runtime checking of service substitutability	N/A	N/A
Yeung [76]	Choreography conformance	WS-CDL, BPEL	N/A
Camara et al. [77]	Behavioral adaptation contract	WSDL, BPEL	MuCRL2
Groefsema et al. [78]	Compliance verification of Web service composition	N/A	NuXMV
Sbai et al. [81]	Compatibility	BPEL	NuSMV
Huynh et al. [79]	Web service similarity	N/A	OnTheFly WSCV
Bataineh et al. [80]	Compliance verification of Web service composition	BPEL	MCMAS

TABLE V
SUMMARY OF DIFFERENT STUDIES ON WEB SERVICE COMPLIANCE AND EQUIVALENCE VERIFICATION USING MODEL CHECKING

E. Web Services Non-functional Requirements Verification

Non-functional requirements of Web services are mainly concerned with Quality of Service (QoS) and security related properties such as privacy, authorization, access control, etc. For example, a requirement specification formula which seeks verification of non-functional properties is specified as follows: $\neg Min \implies P_{\leq 0.20}(\neg Min \cup^{[15,infinite)} Min$. This requirement specification states that the probability to need more than 15 time units to recover from insufficient QoS is at most 20%. In order to detect the potential insecure information leakage, Nakajima [82] presented an idea of a lattice-based security label into BPEL. Nakajima, further, analyzed both the safety and security aspects in a single framework using the model-checking based verification techniques. Schlingloff et al. [83] presented an integrated technique for modeling and automated verification of Web service composition. Their modeling was based on Petri net and they employed model checking technique with alternating temporal logics for correctness verification. Mongiello and Castelluccia [84] presented a formal framework for modeling and verification of business processes written in BPEL. In this framework, modeling is done by translating a given BPEL document into the SMV language and verification is done automatically using NuSMV. In order to address the authorization requirements, Xiangpeng et al. [85] proposed a verifiable framework that integrates Role Based Access Control (RBAC) into BPEL. Further, with the help of model checking, they presented their verification technique.

In order to protect a session between Web services and a SOAP message by the derivation of keys, Xiaolie and Lejian [86] presented a secure session protocol using *WS-Trust* specification and *WS-SecureConversation* specification. Further, the security of the protocol was verified by using a security analysis tool AVISPA. To specify web services composition behavior and property, Zhang et al. [87] used a temporal logic based software architecture description language XYZ/ADL. Then, they translated the temporal logic description to timed-automata and applied refinement checking to verify the correctness of web service composition. Kasse and Mokdad [88] proposed a stochastic comparison based algorithm to check formulas with rewards on multidimensional Continuous Time Markov Chains (CTMC) for composite Web services.

Zhao et al. [28] proposed an approach to generate Web service and BPEL test suit using model checking. The approach takes the advantage of model checking to verify BPEL script at the logical level, and to generate test suit automatically based on the model description, and finally to select test cases with respect to the counterexamples of model checking output. The approach contributes a set of algorithms and its implementation to support a translation from BPEL to LOTOS, and LTS to TTCN behavior tree. Gao et al. [89] studied the monitoring issue of Web service and proposed a predictive service monitoring approach to ensure availability of high-quality Web services at run-time. Considering functional and non-functional requirements, Gao et al. [89] used a probabilistic model checking technique to quantitatively verify the interactive behavior of Web services.

Lu et al. [90] proposed a model checking based approach to verify the satisfiability of behavior-aware privacy requirements in service composition. They extracted LTL specification from the behavior constraints of privacy requirements and they modeled the behavior of BPEL process by extending the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

interface automata, which supports privacy semantics. Then, it is transformed to PROMELA description and verified using SPIN. Jingjing et al. [91] proposed a timed automata based web service composition model. They provided a web service interface description language and a composition automation engine. Further, they validated its performance by using UPPAAL. Rebai et al. [92] proposed a formal verification approach using SPIN model-checker. The approach automatically transforms WS-CDL choreography specifications to PROMELA code and verifies non-functional properties that are written in LTL. Rui-Min and Xiao-Bin [93] proposed a Web service model diction algorithm of non-centralized automaton based on satisfiability modulo theories. In the proposed approach, SMT was used for bounded model checking of timed automaton, and the timed automaton model was directly converted into logical formulas identifiable by SMT.

Mi et al. [94] proposed a method to verify the reliability of BPEL processes. They extracted the model from a BPEL process and analyzed it through probabilistic model checking with PRISM. During the extension process, they added reliability attribute to each invoked sub-services. By structure extraction, the BPEL process was transformed to a PLTS system. Then, they generated a suitable analysis Markov model according to the feature of the PLTS model. Finally, they employed PCTL formula to describe the properties of the system, and verified it with PRISM. El Kassmi and Jarir [95] proposed a model checking based verification technique for Web service security requirements. This technique consists of modeling some security requirements using Finite State Machine (FSM), and their dynamic integration with expressed functional requirements. They used UPPAAL to validate the formalization and integration approach in web service composition. El-Qurna et al. [96] proposed a model checking framework for service trust behaviors. They devised deterministic Push Down Automaton (PDA) based trust behavior model. This model is built based on the observations' sequences, which are derived from the interactions with services. Further, they expressed the regular and non-regular trust behavior properties using Fixed point Logic with Chop (FLC) and verified the properties using a symbolic FLC model checking algorithm. Table VI presents a summary of different studies on non-functional requirements verification of Web services using model checking.

Studies	Verification Goal	WS Technology	Model Checker
Nakajima [82]	Safety and security	BPEL	SPIN
Schlingloff et al. [83]	Usability, composability, and abstraction	BPEL	N/A
Mongiello and Castellucia [84]	Reliability of the business process design	BPEL	NuSMV
Xiangpeng et al. [85]	Role Based Access Control (RBAC)	BPEL	SAL
Xiaolie and Lejian [86]	Secure session protocol	N/A	AVISPA
Zhang et al. [87]	Refinement checking of Web service composition	N/A	UPPAAL
Kasse and Mokdad [88]	Quality of Service (QoS)	N/A	N/A
Zhao et al. [28]	Web service reliability verification	BPEL	N/A
Gao et al. [89]	Reliability prediction of Web services	N/A	N/A
Lu et al. [90]	Web service behavior-aware privacy requirements	N/A	N/A
Jingjing [91]	Feasibility of automatic web service composition	N/A	UPPAAL
Rebai et al. [92]	Non-functional properties	WS-CDL	SPIN
Rui-Min and Xiao-Bin [93]	Optimization	N/A	N/A
Mi et al. [94]	Reliability	BPEL	PRISM
ElKassmi and Jarir [95]	Privacy, integrity, authorization	N/A	UPPAAL
El-Qurna et al. [96]	Service trust behavior	N/A	MCFLC

TABLE VI
SUMMARY OF THE WORKS ON NON-FUNCTIONAL REQUIREMENT VERIFICATION OF WEB SERVICES USING MODEL CHECKING

F. Web Services Time Requirement Verification

Time requirements could be considered as a specific category of non-functional requirements. However, in this subsection, 'time' stands for temporal properties [97] of the Web services. It should not be confused with the time requirements in non-functional requirements such as response time, mean time to failure, etc. Let us consider an example of 'Internet purchase process' (presented in [98]). There are three actors in this example: a customer, a seller and a carrier. The Internet purchase works as follows: A customer wants to buy a product by using Internet. There are several sellers that offer different products. The customer contacts a seller in order to buy the desired product. The seller checks the stock and contacts with a carrier. Finally, the carrier delivers the product to the customer. Following requirement specification represents the situation that if a customer is waiting for an order then the carrier must pick up the order on time and must be delivered within 24 hours. The interval to deliver the product is the time that the seller has stipulated, one day, which is the main temporal constraint. This example situation can be represented by the following requirement specification: $Customer.WaitOrder \rightarrow (Carrier.PickUp \wedge Clock_{deliver} \leq 24hours)$.

Díaz et al. [98] presents an approach for analyzing and verifying the time requirements of Web service compositions via goal-driven models and model checking techniques. They employed a goal-driven model that was an extension of the goal model KAOS. The goal model specified the properties that the system must satisfy and the UPPAAL was used to verify the model. Kazhamiakin et al. [99] proposed a formal technique to capture and verify Web service compositions time related properties, where compositions are modeled using BPEL4WS. They defined a timed state transition system for Web services, namely WSTTS that can capture the timed behavior of web services in a composition scenario. Díaz et al. [50] presented an analysis technique for the timed behavior

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

of Web services. Their technique is based on the translation of service description into the timed automata, and then, they used UPPAAL tool to simulate and analyze the system behavior.

Martinez et al. [100] proposed a technique for the design and verification of time requirements about Web services compositions by using the Web services translation tool. Using a timed automata representation, they demonstrated how to design a desired system and the verification of some properties on it. Mei et al. [21] proposed a formal technique to verify the time related properties of Web service compositions (modeled as BPEL4WS processes). They defined a timed automata (namely WS Timed Automata) for Web services that can capture the timed behavior of the web service compositions. The generated time automata was verified against the requirements using UPPAAL. Cambronero et al. [101] proposed a technique for the validation and verification of composite Web service systems with timing restrictions. They defined an operational semantics for a relevant subset of the WS-CDL. A translation of the considered subset of WS-CDL into a network of timed automata. They used the UPPAAL for the validation and verification of the described system, by using the generated timed automata.

Rawand et al. [102] presented a formal technique for modeling and verification of Web service compositions, where Web services were modeled by using open Workflow Nets (oWF nets). They verified the temporal requirements (written in CTL) against a Web service composition model (written as a SMV code) by using NuSMV. Table VII presents a summary of different studies on Web services time requirements verification using model checking.

Studies	Verification Goal	WS Technology	Model Checker
Díaz et al. [98]	Time requirements	WS-CDL	UPPAAL
Kazhamiakin et al. [99]	Time requirements	BPEL	N/A
Díaz [50]	Times behavior of Web services	WSCI-WSCDL	UPPAAL
Martinez et al. [100]	Time restrictions in service composition	N/A	UPPAAL
Mei et al. [21]	Time properties	N/A	UPPAAL
Cambronero et al. [101]	Composite Web services with timing requirements	WS-CDL	UPPAAL
Rawand et al. [102]	Temporal properties	N/A	NuSMV
Zhao et al. [103]	Timed behavior of choreography	N/A	PAT

TABLE VII
SUMMARY OF DIFFERENT STUDIES ON WEB SERVICE TIME REQUIREMENT VERIFICATION USING MODEL CHECKING

G. Multiple Verification Goals

In this subsection, we review the model checking based Web service verification approaches that focus on the verification goals belonging to more than one classes. Nakajima [104] identified faulty flow descriptions (written in WSFL) by the use of software model-checking technology. The properties checked were reachability, deadlock-freedom, or application specific progress properties. LTL was used for expressing the application specific properties. Betin-Can et al. [105] presented a verifiable design pattern and based on the pattern, they presented a modular verification approach to check the global behavior of services, their interfaces, and conversation among the services. Xiangpeng et al. [106] proposed a language CDL as a formal model of the simplified WS-CDL. Further, they translated WS-CDL to the input language of the model checker SPIN, which verifies the correctness of a given choreography. Dìaz et al. [113] presented a technique for the correct generation of WS-BPEL skeleton documents from the WS-CDL documents by using the timed automata and UPPAAL model checker. Vaz and Ferreira [107] proposed the use of model checking for the verification of Web service business process behavior and workflow patterns by translating the patterns into PROMELA.

Hongli et al. [108] proposed a choreography language CDL by which they verified control-flow properties and channel-passing related problems. Quyet et al. [109] presented a business process verifying technique to translate BPEL processes into PROMELA programs by using the Labeled Control Flow Graphs (LCFGs). Gao et al. [110] proposed a method for modeling the behavior of atomic services by using the interface automata and verifying the requirement specifications in a quantitative way. Ibrahim and Khalil [111] defined a set of transformation rules to generate a model that can be verified using the model checker UPPAAL. Huynh et al. [112] proposed a collection of approaches to address the problem of state-space exploration in the process of model checking of Web services. Their proposed approaches include a LTS-based model, heuristics strategies to find the best potential composition, and a bitwise indexing mechanism for fast location of suitable Web services. Table VIII presents a summary of different studies on Web services model checking having multiple verification goals.

H. Other Verification Goals

In this subsection, we review the model checking based Web service verification works whose verification goals do not belong to any of previously mentioned verification classes. Verification of Channel Passing in Choreography is an example of unclassified category. People have different viewpoints about the concept of channel. We stipulate that a channel is bound to a special service, its messages receiver. In some circumstances, it might be called port or address. By channel passing, it is meant that a service knowing a channel can pass the channel information to another service, and then the latter can send messages via the channel to the receiver.

Lomuscio et al. [114] proposed a multi-agent based approach to model check Web service composition. Along with Web services safety and liveness properties, they also considered epistemic properties for the analysis and verification purpose. Their approach was able to capture the epistemic (knowledge of processes) modalities in addition to the temporal modalities and was also able to reason about these modalities. Wieczorek et al. [115] proposed Model-Based Integration Testing (MBIT) methods for the service choreographies called Message Choreography Models (MCM). Further,

Studies	Verification Goal	WS Technology	Model Checker
Nakajima [104]	Web service flow description and interaction verification	WSFL	SPIN
Betin-Can et al. [105]	Web service behavior, interface, and conversation verification	BPEL	SPIN
Xiangpeng et al. [106]	Semantics of WS-CDL	WS-CDL	SPIN
Díaz et al. [113]	Correctness of the services, time constraints, interaction and deadlock	WS-CDL, BPEL	UPPAAL
Vaz and Ferreira [107]	Workflow and business process behavior	N/A	SPIN
Hongli et al. [108]	Control-flow properties and channel-passing related problems	N/A	SPIN
Quyet et al. [109]	Business process control-flow and service interaction	BPEL	SPIN
Gao et al. [110]	Atomic service behavior and performance	N/A	N/A
Ibrahim and Khalil [111]	Control-flow and non-functional properties	BPEL	UPPAAL
Huynh et al. [112]	Functional properties and QoSs	N/A	PAT

TABLE VIII
SUMMARY OF DIFFERENT STUDIES ON WEB SERVICES MODEL CHECKING WHERE MULTIPLE VERIFICATION GOALS ARE THERE

MCMs were translated into Event-B models and provided as input to the test suit generator which used the model checker ProB. Peng et al. [116] proposed a model checking based verification approach for the channel passing related problems in choreography. They defined a WS-CDL based choreography language, namely $Chor_c$, in which basic service interaction patterns are represented by using pre and post conditions. Further, for the verification purpose, a $Chor_c$ description is translated into a model acceptable by the SPIN model checker.

Kang and Wang [117] extended WS-CDL with an XML-based new formal language, namely TLA4CDL, that is based on the concept of temporal logic of actions. The prime advantage of TLA4CDL is that it can capture action-action and action-state relationship which is not possible with only temporal modalities enabled logics such as LTL. Further, they proposed a model checking algorithm that was based on the concept of TLA4CDL. Fang et al. [118] presented an approach to automatically discover the suitable services and compose them by using the concept of bounded model checking. Particularly, given a set of available Web services and a requirement specification (written in LTL), the approach discovers the services that satisfies the requirement specification. In case, if no such suitable service is available, it composes the available services to realize the specification by keeping the bound (say K) in mind. Rossi [119] proposed a model checking and logic based algorithm for the verification of service compositions. The algorithm was filter (that prevents service misbehavior) based and intended for multileveled service compositions. Her focus was to verify security and correctness requirements written as security policies that are dynamically defined by the service participants or users.

Lomuscio et al. [120] proposed a verification technique for contract-regulated service compositions. They specified Web services and the governing contracts as WS-BPEL behaviors. Further, in order to verify the compliance or violation of contracts they employed an extended version of temporal-epistemic logic. Yongxiang et al. [121] proposed a modeling and verification approach for the composition of cloud manufacturing services. Their proposed technique was based on Unified Modeling Language (UML) and a process algebra based Web service orchestration calculus, namely COWS. A service composition modeled as activity diagram is converted into COWS syntax and considered as an input for the verification process. Kokash and Arbab [122] proposed a formal framework for modeling and verification of Long-Running Transactions (LRTs). They modeled and verified the LRTs by using a channel-based coordination language, namely Reo. By assuming that participant entities do not know the existence of each other in advance, with the help of Reo, a designer can analyze and verify the transactional behaviors of the system before the systems actual implementation. Bourne et al. [123] proposed a modeling and verification technique that captures and verifies the transactional properties (component and composition level) of composite Web services. Here, a user can specify the transactional requirements using temporal logic constructs, later, their conformance is verified using model checking based techniques. Table IX presents a summary of different studies on Web services requirements verification using model checking where goals are not classified.

Studies	Verification Goal	WS Technology	Model Checker
Lomuscio et al. [114]	Temporal and epistemic properties	N/A	MCMAS
Wieczorek et al. [115]	Integration testing	N/A	ProB
Peng et al. [116]	Channel passing in choreogrpay	WS-CDL,BPEL	N/A
Kang and Wang [117]	Temporal and action attributes	WS-CDL	N/A
Fang et al. [118]	Service discovery	N/A	N/A
Rossi [119]	Non-interference property, deadlock, livelock	N/A	N/A
Lomuscio et al. [120]	Contract regulated Web service composition	BPEL	MCMAS
Yongxiang et al. [121]	Cloud manufacturing services	N/A	CMC
Kokash and Arbab [122]	Long-running transactions	WSDL, BPEL	mCRL2
Bourne et al. [123]	Transactional requirements of Web service compositions	N/A	NuSMV

TABLE IX
SUMMARY OF DIFFERENT STUDIES ON WEB SERVICE REQUIREMENT VERIFICATION USING MODEL CHECKING WHERE GOALS ARE NOT CLASSIFIED

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

IV. Analysis of SLR Results

Analysis of this SLR on model checking of Web services is made addressing the research questions discussed in Table I. We investigated the scalability and broad applicability of the surveyed systems based on their proposed methodologies. For that, we selected the papers which provide details of their methodologies and discuss the implementation. Since the objectives of the surveyed systems are not exactly same, it was not possible to compare them based on a common benchmark. A model checking process consists of three phases: modeling the system, requirement specification, and verification technique. We grouped the papers into three categories considering their focus on one or more phases of the model checking process.

1) Applicability of the surveyed systems that contributed to the modeling aspect. Web services are modeled using BPEL, WS-CDL, etc. However, model checking accepts a model that is written in its own modeling language such as SPIN or NuSMV. Therefore, a Web service model should be extended to the model acceptable for a model checking technique. For a model extension, there are generally two methods. One is modifying a model to obtain another model that is appropriate for another purpose; the other is using the concept of annotation to refer to pieces of information that are not necessary for determining the behavior of a model. Dai et al. [12] augmented BPEL4WS models with related constraints to raise the reliability of a Web service flow. An underlying BPEL4WS model and one or more constraint annotation layers compose a complete specification of a business process imposed specific constraints. Theoretically, this proposal is very useful as it has additional features if compared with other contemporary approaches. However, the proposed tool MCT4WS lacks full implementation and is not made publicly available. The implementation proposed in [11] takes PHP5 programs as input, and verifies joint properties of these programs running concurrently. The main challenge when applying formal methods to scripting languages such as PHP is to model the extensive library. Sharygina et al. [11] completed that task to the point that was required for the property described in the paper to pass; a verification tool of industrial relevance has to include an almost complete set of models for all functions offered by the execution environment. Dong et al. [16] (similarly [13]) established a formalized analysis model, provided specifications and semantics of business flow, and translated it into the asynchronous communication model of SPIN. To illustrate the effectiveness and efficiency of their proposed method, they analyzed and verified the SAS protocol as well as the Loan and Auction protocols written in BPEL. Based on their experimental results, the model can capture the attributes of control flow and data flow of BPEL and can be easily extended to Multi-Agent System (MAS) model to analyze the dynamic Web services. Nakajima [82] introduced an idea of a lattice-based security label into BPEL in order to detect potential insecure information leakage. Although the proposed approach is applicable, security labels used in the paper are simple; it would be more beneficial if other high-level security policies were encoded for a variety of purposes.

Corredini et al. [14] analyzed the WS-CDL language source and isolated the necessary information to build a Promela model suitable for the verification using the SPIN model checker. However, no proof was given regarding the validity of the mapping and of the approach. Kang et al. [117] proposed TLA4CDL that was an expressive and easy-to-use formal language used for describing the verifying attributes of WS-CDL choreographies. As is obvious from the results of the experiments, WS-CDL model checker based upon TLA4CDL is capable of verifying specified attribute like deadlock, safety, liveness, and fairness, and is able to check WS-CDL choreographies composed of a number of participants. The experiments also show that applying the partial order method will improve its checking performance by greatly reducing the searching state space size. Xu et al. [19] proposed a formal approach to check Web service models by formally extracting the WS-CDL based specifications in a communicating sequential process. The proposed approach avoids the possibility of misunderstanding and different implementations and provides a formal semantics to WS-CDL. Hallé [36] proposed the automated generation of Web service stubs based on temporal logic specifications. Preliminary experimental results indicate that the use of a model checker as a back-end engine to produce message traces compliant with these formulae is feasible. Despite a potential exponential growth in execution time with respect to domain size, for modest domain sizes, it achieves response times appropriate for its use as a development and testing tool for third-party applications. Further, an extended version of the stub mechanism described in this paper was not completely developed. Ravn et al. [41] proposed a formal approach to analyze Web service protocol behaviors (Web service business activity standards) and to verify general correctness criteria. However, a simple relaxation of the perfect communication policy results in incorrect behavior of the proposed protocol. Kokash and Arbab [122] proposed Reo which is able to deal with tricky transactional patterns (e.g., discriminator choice) easily. Due to its combination of synchrony and asynchrony, Reo is more suitable for specifying exception and compensation handling in Long Running Transactions (LRTs) than classical Petri-nets. On the other hand, Reo is easier than process algebras, which makes it a promising technique for practical applications for designers without having strong formal background. Furthermore, by introducing new channels (e.g., data transformers) in Reo and appropriate constraints in CA, we can deal with formal verification of various process properties including data related constraints and non-functional requirements. Díaz et al. [39] presented a generic model for publishing and managing Web service resources in the context of Web services with distributed resources. The model includes operations for clients to discover and subscribe to resources, with the intention of being notified when the resource property values fulfill certain conditions. This model can be modified by extending the parameters used in the discovery process by adding information regarding the service response latency, the service quality, the price per use, the payment policy and so on. This extension can be easily accomplished by modifying the publishers so they provide this

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

new information to the repository that would be stored for future use during the discovery process. El-Qurna et al. [96] proposed a service trust behavior model, represented as a finite transition system that explores all the possible trust behaviors of the service. The trust behavior algorithm is affected by both the number and the length of the observations sequences in the dataset. Using multiple regression analysis, one can observe that the length of the observations sequences has more impact on the execution time of the trust behavior algorithm than the number of the trust sequences in the given dataset. Moreover, the experiments show that the non-regular trust behavior properties need more time compared to the regular trust behavior properties. The main limitation of non-regular behaviors verification is its EXPTIME complexity.

Some researchers claim that BPEL and WS-CDL are not informative enough to be used for model checking. Therefore, either they modified the representation or proposed different types of translation/mapping schemes, such as BPEL to Promela/SPIN [33], [27], BPEL to CSP [35], [76], BPEL to LOTOS [28], BPEL to Process Algebra Nets [72], BPEL to Symbolic Transition System [37], BPEL to guarded automata [44], extended and marked BPEL to an automata model [80], translation of WS-CDL specifications into a network of timed automata [101], WS-CDL into CSP [76]. These translation schemes are applicable for realistic-size case studies. Further, solutions proposed in [37], [27] would work better if a language would have been defined based on workflow (such as Business Process Modeling Notation (BPMN)) to express the test purposes.

2) Applicability of the surveyed systems that contributed to the requirement specification aspect. Barker et al. [18] proposed a Multiagent Protocols (MAP) Web service choreography language. MAP supports multiparty communication between an unknown and unbounded set of participants, allowing multiple MAP protocols to be executed concurrently. These multiparty scenarios are very common, as demonstrated by the Service Interaction Patterns. In contrast, WS-CDL does not directly support parallel conversations with an unknown number of participants. Schlingloff et al. [83] devised that the notion of usability can be formulated in alternating-time temporal logic ATL. They presented the results concerning computer aided usability analysis for acyclic workflow modules. The analysis covers scenarios of central as well as distributed usability. The given results established that the usability problem was closely related to the model checking problem for alternating temporal logic. Further, investigations suggest that generalizations of the proposed usability criteria are also applicable to cyclic web services. This would enable one to verify a larger class of web services. Xiangpeng et al. [106] defined a small choreography language CDL, which can be viewed as a subset of WS-CDL. It models choreography with a set of participant roles and the collaboration among them. However, the opaque points or inconsistencies in the language definition were possible. Wan et al. [56] defined and implemented the mapping rules for transforming the agent hypergraphs and CTL^{*CA} specifications into the ISPL language. They implemented the mapping rules and the PNAWS case study scenario along with the specifications in ISPL and verified them with MCMAS. Experimental results for the implementation clearly showed that the state space grew exponentially, but thanks to symbolic model checking the execution time was low. Camara et al. [77] presented an interactive approach which speeds up the contract specification process and reduces the risk of mistakes in the specification. This approach relies on compositional and hierarchical techniques, a graphical notation, and different verification and validation techniques. Experimental results show a reduction both in the amount of effort that the designer has to put into building the contract, as well as in the number of errors present in the final result. El Kholy et al. [58] proposed a branching-time temporal logic, called CTL_{cc}, which extends CTL with temporal modalities for commitments and their fulfillments. The proposed approach is applicable and easy to implement; commitments can be modeled as abstract data types that can be treated by propositions in CTL model, and objects with different states can be encoded as SMV modules. However, the approach does not reflect the real and concrete meanings of commitments; and no formal semantics is given to commitments.

3) Applicability of the surveyed systems that contributed to the verification technique aspect. Some researchers have modified the model checking techniques slightly as the existing model checkers have only very limited support for the programming languages and communication mechanisms used by typical implementations of web services ([11]). Yu et al. [124] proposed a scalable modular verification technique for composite web services written in BPEL. Based on the experimental results, this approach reduces the number of states significantly by decomposing the verification task. Bersani et al. [75] introduced a verification technique, based on Satisfiability Modulo Theories (SMT), for an extension of Propositional Linear Temporal Logic with Both past and future operators (PLTLB). The implemented prototype was shown to be considerably faster and with smaller memory footprint than existing ones based on traditional BMC, due to the conciseness of the problem encoding. Their encoding was found very suitable for application to a real problem taken from the SOA domain and showed better performances and lower memory occupation than the other available encodings. Groefsema et al. [78] proposed a mapping of service compositions to Kripke structures by using colored Petri nets. If a model does turn out to be too large for model checking, this approach allows splitting formulas into multiple sets, each resulting in a much smaller reduced Kripke structure. Although, in worst case, the proposed technique is responsible for a state explosion, this is of little concern for compositions with average and even large parallel areas. Foster et al. [43] proposed a trace equivalence based verification process. To perform trace equivalence of the prepared Finite State Process notation (FSP) specifications, two checks are performed. The first checks if the BPEL4WS FSP provides possible traces that the MSC FSP cannot. The second checks if the MSC FSP provides possible traces that the BPEL4WS FSP cannot. This technique worked as an alternative to possible world based verification process. Bentahar et al. [60] (similarly [59]) discussed how composite Web services could be designed, modeled, and verified based on their control and operational behaviors. Based on

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

the experimental results the proposed system could perform automated checking to detect service design problems. They performed their experiments with only 10 services, therefore, one needs to conduct more experiments to study performance and usability of the system. Gao et al. [89] employed a probabilistic model checking to monitor Web services both at functional and nonfunctional verification levels, and used an unary linear regression analysis method to predict the reliability of Web service. Their experimental results showed that the real service failure may occur before the predicted time point. Thus, verifier should consider the availability of service-oriented software when the prediction method is employed. These experiments also demonstrated that frequently performing probabilistic model checking was not an effective solution to monitor Web services. Kazhamiakin et al. [45] (similarly [48]) proposed a verification approach, based on explicit state model checking and SPIN. The approach was based on a novel, compositional encoding of the LTL constraints that defined the valid behaviors of the requirements model in the verification tasks. Experimental results provided in [48] proved that the approach was viable, even if the performance was a rather serious limit for its applicability.

Table X presents applicability of different model checking techniques in the context of Web services verification. It also provide some considerations on what to use and when based on our survey. In Table X, each tool corresponds to a different flavor of the model checking and they differ from one another mainly in their verification focus. By analyzing the distribution of the articles based on their use of model checking tools, we observe that 27% of our selected articles have not been used any model checking tools for their proposed model checking based Web service verification approaches. A possible reason behind this fact could be that the model checking technique is not applicable in the case of Web service verification as it is. In most of the cases, based on the primitive characteristics of Web services, some modifications has to be done to the classical model checking techniques. If one does not modify the model checking approach then she has to resort to intermediate representations and/or other modeling and verification techniques. For instance, Table XI shows that in many of the collected articles, other modeling and verification techniques are also used along with a model checking technique. Moreover, we observe that in earlier years, model checking alone was used for verification, whereas in recent years the focus has shifted towards hybrid approaches to achieve more expressive and reasoning power.

V. Research Implications and Future Directions

A. Advantages for Practitioners and Researchers

This SLR presents a methodical and comprehensive study of 105 most relevant model checking based verification approaches for Web services. By using this SLR, practitioners and researchers can explore to what extent existing research has made progress towards model checking of Web services, thereby, reducing the time and complexity of searching and investigating the available studies. Moreover, by using this SLR, they can find implications for practice and policy, gaps, and inconsistencies in the literature, and possible future research directions.

B. Research Challenges and Future Directions

Based on our analysis of the research works considered in this SLR, we found that the following points can be considered for the possible future research directions.

1) Related to Model Checking in the Context of Web Services Verification

- Abstractions such as *Counter Example Guided Abstraction Refinement* (CEGAR) and *Predicate Abstraction* have been employed to make model checking more efficient for the large systems of Web services [11], [15], [38]. However, abstraction refinement require further exploration and predicate abstraction has to be developed in such a way that it could support the verification of liveness as well as safety properties.
- Though state space reduction techniques such as symbolic representations, abstractions, partial order reductions, etc. are available, state explosion is still one of the key problems that hinders application of model checking on very large set of Web services [85].

2) Related to Web Service Standards in the Context of Web Services Verification

- Verification of the specification properties has to be dealt in more detail and for the detailed verification of properties, a BPEL or WSDL document should be annotated with some extra information such as constraints.
- A verification technique should be well suited for both orchestration and choreography.
- Concurrency issues in Web service compositions are very common such as *Flow* construct in BPEL. They need special attention.
- Several Web service standards possess serious limitations that have to be overcome, like WSCL specifies message sequences between only two participants. Moreover, these standards should support error handling, transaction management, etc.
- In many research works, intermediate representations are used; at later stage, which are translated into the input language of a model checker such as BPMN models to PROMELA [107], WS-CDL to CDL [108], LCFG to PROMELA [109], BPEL4WS to Petri nets [83], etc. This translation process should be reduced or optimized and should be provided with the automatic translation facility.
- There should be support for the verification of hierarchy of choreographies.

3) Related to Automation and Dynamic Composition in the Context of Web Services Verification

- Web services can be easily and transparently substituted in the case of static compositions, whereas it is difficult in the dynamic compositions.
- There should be techniques for verifying the compatibility between two choreographies where the involved services are not known in advance.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2845401, IEEE Transactions on Services Computing

Classification Criteria	Underlying Technique	When to Use (in the Context of Web Services)	Supported Logic/ Representative Tools
Model checking techniques with different types of state space representations	Explicit state space representation	For checking safety and liveness properties of Web services if the generated number of states are not high.	LTL or Buchi Automata/ SPIN [127]
	Symbolic state representation using BDD technique	If the generated number of states are very high (astronomical no of states).	LTL or CTL/ SMV, NuSMV [128], NuXMV, etc.
	Bounded model checking	It is used to find the more shallow bugs quickly (It searches for a counterexample in executions whose length is bounded by some integer k).	CBMC[129],EBMC
	Counterexample-guided Abstraction Refinement	It is used to map the classes of similar states into single abstract states. It reduces model size drastically. However, invalid counterexamples are possible.	SLAM/MAGIC, BLAST [130], BLADE, etc.
Real-time model checking technique	Timed automata	Applicable to the Web service systems that can be modeled as a collection of non-deterministic processes with finite control structure and real-valued clocks, communicating through channels or shared variables.	TCTL/ UPPAAL [131]
Probabilistic model checking techniques	Discrete Time Markov Chain (DTMC)/ Continuous Time Markov chain (CTMC)/ Probabilistic Automata (PA)	For dependability analysis/ for establishing if a desired property holds in a probabilistic mode	PCTL, Continuous Stochastic Logic (CSL), / PRISM [132]
Autonomous systems model checking technique	Multiagent system theory	Useful if web services are realized as a number of interacting intelligent computational entities	ISPL/MCMAS [133]
Process-based model checking technique	Process algebra (CCS and/or CSP)	For verifying Web service systems modeled and specified as CSP Processes, for assertion checking, and for compatibility/equivalence checking	CSP/ FDR4, TAPAs, PAT [134]
Petri net based model checking technique	Petri net	Useful if concurrency related issues are more concerned among available Web services	CTL*/LoLA [135]

TABLE X
APPLICABILITY OF DIFFERENT MODEL CHECKING TECHNIQUES IN THE CONTEXT OF WEB SERVICES

Key Studies	Supplementary Technique	Verification Goal Classification
[35], [11], [19], [103]	CSP	Behavioral, concurrency, composition, time requirements
[30], [28]	LOTOS	Composition, Non-functional
[72][18]	π-calculus	Equivalence, composition
[83]	Petri net	Non-functional
[78]	CPN	Equivalence
[125]	TCPN	Composition
[136], [137], [117]	TLA	Composition, behavioral
[56], [49], [76], [121]	UML	Interaction, equivalence
[102], [40], [81]	oWF	Time requirements, behavioral, equivalence
[109]	LCFG	Combined
[82]	Lattice	Non-functional
[75], [37], [93]	SMT	Equivalence, behavioral, Non-functional
[31]	Event calculus	Composition
[138]	PDDL	Concurrency

TABLE XI
OTHER TECHNIQUES USED WITH MODEL CHECKING

- While verifying automated Web service composition, There should be filtering techniques to exempt unrelated services in the discovery and composition process.
- Model checkers such as NuSMV, UPPAAL, etc. should support dynamic verification of Web service composition.
- Since execution is controlled in static compositions, modeling and verification of fault handling mechanisms are easier, whereas it is difficult in dynamic compositions.
- There should be support for automatic extraction of interface expressions from abstract BPEL processes as well as from WS-CDL choreographies.

4) Related to Equivalence Verification in the Context of Web Services Verification

- Equivalence checking or service matchmaking should not be done only based on the interface matching or input-output compatibility, their behavior and contribution to a composite service should also be considered.
- The conformance problem between orchestration and choreography (for instance, whether a given BPEL orchestration processes is consistent with a given WS-CDL choreography model) requires thorough investigation.

5) Related to Experimentation and Tools in the Context of Web Services Verification

- An integrated modeling, verification, and testing environment, is required. For Web services, at present, no such facility is available.
- Significant importance must be given to counterexample presentation and visualization to guide the developer in the error correction process.
- There are requirements of a standard verification benchmark, set of services, and tools, so that results claimed in a paper could be validated and compared with other works.
- Interactive environment should be there for the graphical specification of temporal properties and for the dynamic evaluation of such properties.
- A Web service verification tool supports either WSDL or BPEL but not both. Tools are required that could support both of the standards in the verification process.

6) Other Possible Future Research Directions

- Optimization is required at several places in the Web service verification process. For instance, some potential optimization techniques should be applied to the test generation steps without sacrificing the goal of minimal test effort and optimized Boolean encoding approaches are required to improve performance of the system.
- There should be support for parallel workflow patterns, enabling more sophisticated guard conditions, and scalability tests with increasingly complex scenarios.
- Verification facility for electronic contract should be available.
- Verification processes for Web services should support Model Driven Architecture (MDA), model transformation, and modular verification.
- In the context of Web services, capabilities of Alternating Temporal Logics (such as ATL) should be more explored for the requirement specification.
- Analyzing and determining the impact of fault tolerance and compensation actions.
- There should be more enriched support for the verification of sessions and multi-party communication.
- Scalability of a verification technique should be thoroughly investigated.
- There should be support for probabilistic investigation of each process in a composition scenario. Though, similar work is available, more precision is required like distribution of probability among constructs.
- Monitoring facility for the execution of composite web services at run-time should be available. Also, run-time conflict detection facility for the services should be available.

C. Threats to Validity

Threats to completeness. In order to conduct a systematic review, a reviewer constructs a search string in such a way that she could collect all relevant studies without spending much effort and time. For this SLR, we searched seven databases and constructed the search string in such a way that it can include all the articles that are anyhow related to model checking and Web services. Further, we refined our initial selection by using the inclusion and exclusion criteria mentioned in Section II. The search string for initial selection was flexible enough to allow all the relevant articles, however, some articles might be missing due to linguistic barriers.

Threats to the selection of studies. Since different researchers may have different views on final selection of the studies, two researchers worked together to avoid the possibilities of any conflicts. However, if even after this a disagreement was persisted, other researchers from the same area were called to make a final decision on the selection of primary studies.

Threats to data extraction. We searched seven databases and collected 582 articles in our initial selection. Among those, 105 articles were finally selected. We validated our search results by cross-checking individual journals and proceedings respectively.

VI. Conclusions

In this paper, we described an SLR on the referred research articles appeared during the period of 2002-2017 that are related to the model checking of Web services. Based on the verification goals, we classified the existing approaches into six classes: composition verification, interaction verification, behavioral verification, equivalence verification, non-functional requirement verification, and time requirement verification. Further, we presented a discussion on this SLR and highlighted some of the possible future research directions.

References

[1] E. M. Clarke, O. Grumberg, and D. Peled, *Model checking*. MIT press, 1999.

[2] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software eng. domain," *Journal of systems and software*, vol. 80(4), pp. 571–583, 2007.

[3] P. Jamshidi, A. Ahmad, and C. Pahl, "Cloud migration research: a systematic review," *IEEE Trans. on Cloud Computing*, vol. 1, no. 2, pp. 142–157, 2013.

[4] Q. Z. Sheng, X. Qiao, A. V. Vasilakos, C. Szabo, S. Bourne, and X. Xu, "Web services composition: A decade's overview," *Inform. Sci.*, vol. 280, pp. 218–238, 2014.

[5] M. Röglinger, "Verification of web service compositions: An operationalization of correctness and a requirements framework for service-oriented modeling techniques," *Business & Inform. Systems Engineering*, vol. 1(6), pp. 429–437, 2009.

[6] N. Milanovic and M. Malek, "Current solutions for web service composition," *IEEE Internet Computing*, vol. 8(6), p. 51, 2004.

[7] R. Huang, Hai & Mason, "Model checking technologies for web services," in *Proc. of the 2nd Int. Workshop on Collaborative Computing, Integration, and Assurance (WCCIA'06)*, 2006, pp. 217–224.

[8] K. S. Khan, R. Kunz, J. Kleijnen, and G. Antes, "Five steps to conducting a systematic review," *Journal of the Royal Society of Medicine*, vol. 96(3), pp. 118–121, 2003.

[9] C. Jatoth, G. Gangadharan, and R. Buyya, "Computational intelligence based qos-aware web service composition: A systematic literature review," *IEEE Trans. on Services Computing*, vol. 10, pp. 475–492, 2017.

[10] F. Leymann, *Web services flow language (WSFL 1.0)*. IBM Academy of Technology, 2001.

[11] N. Sharygina and D. Kröning, "Model checking with abstraction for web services," in *Test and Analysis of Web Services*. Springer, 2007, pp. 121–145.

[12] G. Dai, X. Bai, and C. Zhao, "A framework for model checking web service compositions based on BPEL4WS," in *Proc. of the Int. Conf. on e-Business Engineering*. IEEE, 2007, pp. 165–172.

[13] J.-y. Zhang, F.-c. Yang, and S. Sen, "Detecting feature interactions in web services with model checking techniques," *The Journal of China Universities of Posts and Telecommunications*, vol. 14, no. 3, pp. 108–112, 2007.

[14] F. Corredini, A. De Angelis, and A. Polzonetti, "Verification of WS-CDL choreographies," in *Proc. of the 2nd European Young Researchers Workshop on Service Oriented Computing (YR-SOC 2007)*, 2007, pp. 13–18.

[15] J. Qian, G. Cai, T. Gu, and L. Zhao, "Abstract model checking for web services," *Wuhan University Journal of Natural Sci.*, vol. 13, no. 4, pp. 466–470, 2008.

[16] R. Dong, Z. Wei, and X. Luo, "Model checking behavioral specification of BPEL web services," in *Proc. of the World Congr. on Engineering*, vol. 1, 2008.

[17] X. Luo, Z. Tan, and R. Dong, "Automatic verification of composite web services based on temporal and epistemic logic," in *Proc. of the 3rd Int. Conf. on Genetic and Evolutionary Computing*. IEEE, 2009, pp. 693–696.

[18] A. Barker, C. D. Walton, and D. Robertson, "Choreographing web services," *IEEE Trans. on Services Computing*, vol. 2(2), pp. 152–166, 2009.

[19] D. Xu, Z. Lei, W.-m. Li, and B.-f. Zhang, "Model checking web services choreography in process analysis toolkit," *Journal of Shanghai University (English Edition)*, vol. 14, pp. 45–49, 2010.

[20] X. Luo, J. Lu, K. Su, and R. Dong, "Translation-based verification of web services composition via ZING," in *Proc. of the Int. Conf. on Intelligent Computing and Integrated Systems (ICISS)*. IEEE, 2010, pp. 596–600.

[21] J. Mei, H. Miao, Q. Xu, and P. Liu, "Modeling and verifying web service applications with time constraints," in *Proc. of the 9th Int. Conf. on Computer and Inform. Sci. (ICIS)*. IEEE, 2010, pp. 791–795.

[22] Z. Xinlin, "Categorical description and checking for web services composition model," in *Proc. of the 4th Int. Symp. on Computational Intelligence and Design (ISCID)*. IEEE, 2011, pp. 206–210.

[23] G. Zhang, H. Shi, M. Rong, and H. Di, "Model checking for asynchronous web service composition based on XYZ/ADL," in *Proc. of the Int. Conf. on Web Inform. Systems and Mining*. Springer, 2011, pp. 428–435.

[24] W. Li, Z. Yang, P. Zhang, and Z. Wang, "Model checking WS-BPEL with universal modal sequence diagrams," in *Proc. of the 10th Int. Conf. on Computer and Inform. Sci. (ICIS)*. IEEE, 2011, pp. 328–333.

[25] D. Fu and G. Chen, "A verification method for web services combination based on abstract and refinement technology," in *Proc. of the 2nd Int. Conf. on Consumer Electronics, Communications and Networks (CECNet)*. IEEE, 2012, pp. 119–122.

[26] X. Luo, K. Wang, and F. Wang, "An epistemic model checking approach to web service compositions," *Int. Journal of Wireless and Microwave Technologies (IJWMT)*, vol. 2(6), p. 66, 2012.

[27] V. Todica, M.-F. Vaida, and M. Cremene, "Formal verification in web services composition," in *Proc. of the IEEE Int. Conf. on Automation Quality and Testing Robotics (AQTR)*. IEEE, 2012, pp. 195–200.

[28] H. Zhao, J. Sun, and X. Liu, "A model checking based approach to automatic test suite generation for testing web services and bpel," in *Proc. of the Asia-Pacific Services Computing Conf. (APSCC)*. IEEE, 2012, pp. 61–69.

[29] J. Mei, H. Miao, Y. Chen, and H. Gao, "Verifying web services composition based on interface automata using SPIN," *Int. Journal of Digital Content Technology and its Applications*, vol. 4, no. 8, pp. 23–33, 2010.

[30] H. Zhao, W. Wang, J. Sun, and Y. Wei, "Research on formal modeling and verification of BPEL-based web service composition," in *Proc. of the 11th Int. Conf. on Computer and Inform. Sci. (ICIS)*. IEEE, 2012, pp. 631–636.

[31] E. Zahoor, K. Munir, O. Perrin, and C. Godart, "A bounded model checking approach for the verification of web services composition," *Int. Journal of Web Services Research (IJWSR)*, vol. 10, no. 4, pp. 62–81, 2013.

[32] F. Yu, C. Wang, A. Gupta, and T. Bultan, "Modular verification of web services using efficient symbolic encoding and summarization," in *Proc. of the 16th ACM SIGSOFT Int. Symp. on Foundations of software engineering*. ACM, 2008, pp. 192–202.

[33] S. Nakajima, "Model-checking behavioral specification of bpel applications," *Electronic Notes in Theoretical Computer Sci.*, vol. 151(2), pp. 89–105, 2006.

[34] G. Diaz, M.-E. Cambronero, J. J. Pardo, V. Valero, and F. Cuartero, "Automatic generation of correct web services choreographies and orchestrations with model checking techniques," in *Proc. of the Advanced Int'l Conf. on Telecommunications and Int'l Conf. on Internet and Web Applications and Services (AICT-ICIW'06)*. IEEE, 2006, pp. 186–186.

[35] W. L. Yeung, "Mapping WS-CDL and BPEL into CSP for behavioural specification and verification of web services," in *Proc. of the European Conf. on Web Services (ECOWS)*, vol. 6, 2006, pp. 297–305.

[36] S. Hallé, "Automated generation of web service stubs using ltl satisfiability solving," in *Proc. of the Int. Workshop on Web Services and Formal Methods*. Springer, 2010, pp. 42–55.

[37] L. Bentakouk, P. Poizat, and F. Zaïdi, "Checking the behavioral conformance of web services with symbolic testing and an smt solver," in *Proc. of the Int. Conf. on Tests and Proofs*. Springer, 2011, pp. 33–50.

[38] H. Gao, Y. Zhang, and J. Liu, "Modeling and verifying web service behaviors based on live sequence chart specifications [j]," *Journal of Computational Inform. Systems*, vol. 7(10), pp. 3499–3507, 2011.

[39] G. Díaz, M. E. Cambronero, H. Maciá, and V. Valero, "Model-checking verification of publish-subscribe architectures in web service contexts," in *Proc. of the 30th Annual ACM Symp. on Applied Computing*. ACM, 2015, pp. 1688–1695.

[40] A. B. Azaiez and Z. Sbaï, "Model checking web services choreography," in *Proc. of the 11th Int. Workshop on Enterprise and Organizational Modeling and Simulation, EOMAS*. Springer, 2015, pp. 171–186.

[41] A. P. Ravn, J. Srba, and S. Vighio, "Modelling and verification of web services business activity protocol," in *Proc. of the Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2011, pp. 357–371.

[42] J. Marques, AbinoamP., A. Ravn, J. Srba, and S. Vighio, "Model-checking web services business activity protocols," *Int. Journal on Software Tools for Technology Transfer*, vol. 15(2), pp. 125–147, 2013.

[43] H. Foster, S. Uchitel, J. Magee, and J. Kramer, "Model-based verification of web service compositions," in *Proc. of the 18th IEEE Int. Conf. on Automated Software Engineering*. IEEE, 2003, pp. 152–161.

[44] X. Fu, T. Bultan, and J. Su, "Analysis of interacting BPEL web services," in *Proc. of the 13th Int. Conf. on World Wide Web*. ACM, 2004, pp. 621–630.

[45] R. Kazhamiakin, M. Pistore, and M. Roveri, "Formal verification of requirements using SPIN: A case study on web services," in *Proc. of the 2nd Int. Conf. on Software Eng. and Formal Methods*. IEEE, 2004, pp. 406–415.

[46] C. Walton, "Model checking multi-agent web services," in *Proc. of the AAAI Symp. on Semantic Web Services*, 2004.

[47] X. Fu, T. Bultan, and J. Su, "Realizability of conversation protocols with message contents," in *Proc. of the Int. Conf. on Web Services*. IEEE, 2004, pp. 96–103.

[48] M. Pistore, M. Roveri, and P. Busetta, "Requirements-driven verification of web services," in *Proc. of the 1st Int. Workshop on Web Services and Formal Methods (WSFM'04)*. Elsevier, 2004, pp. 95–108.

[49] H. Cao, S. Ying, and D. Du, "Towards model-based verification of bpel with model checking," in *Proc. of the 6th IEEE Int. Conf. on Computer and Inform. Technology (CIT'06)*. IEEE, 2006, pp. 190–190.

[50] G. Díaz, J. J. Pardo, M. Cambronero, V. Valero, and F. Cuartero, "Verification of web services with timed automata," *Electr. Notes in Theor. Comput. Sci.*, vol. 157(2), pp. 19–34, 2006.

[51] Y. Zheng, J. Zhou, and P. Krause, "A model checking based test case generation framework for web services," in *Proc. of the 4th Int. Conf. on Inform. Technology (ITNG '07)*. IEEE, 2007, pp. 715–722.

[52] A. Deutsch, L. Sui, and V. Vianu, "Specification and verification of data-driven web applications," *Journal of Computer and System Sci.*, vol. 73, no. 3, pp. 442–474, 2007.

[53] Z. Gu, J. Li, J. Tang, B. Xu, and R. Huang, "Verification of web service conversations specified in wscl," in *Proc. of the 31st Int. Computer Software and Applications Conf. (COMPSAC'07)*, vol. 2. IEEE, 2007, pp. 432–437.

[54] P. Parizek and J. Adamek, "Checking session-oriented interactions between web services," in *Proc. of the 34th Euromicro Conf. Software Eng. and Advanced Applications*. IEEE, 2008, pp. 3–10.

[55] G. Shegalov and G. Weikum, "Formal verification of web service interaction contracts," in *Proc. of the IEEE Int. Conf. on Services Computing (SCC'08)*, vol. 2. IEEE, 2008, pp. 525–528.

[56] W. Wan, J. Bentahar, and A. B. Hamza, "Modeling and verifying agent-based communities of web services," in *Proc. of the Int. Conf. on Industrial, Eng. and Other Applications of Applied Intelligent Systems*. Springer, 2010, pp. 418–427.

[57] L.-l. Qian and Y.-h. Chen, "Generating test case specifications of web service composition using model checking," *Journal of Shanghai University (English Edition)*, vol. 15, pp. 409–414, 2011.

[58] W. El Kholy, J. Bentahar, M. El Menshawy, H. Qu, and R. Dssouli, "Modeling and verifying choreographed multi-agent-based web service compositions regulated by commitment protocols," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7478–7494, 2014.

[59] Q. Z. Sheng, Z. Maamar, L. Yao, C. Szabo, and S. Bourne, "Behavior modeling and automated verification of web services," *Inform. Sci.*, vol. 258, pp. 416–433, 2014.

[60] J. Bentahar, H. Yahyaoui, M. Kova, and Z. Maamar, "Symbolic model checking composite web services using operational and control behaviors," *Expert Systems with Applications*, vol. 40, no. 2, pp. 508–522, 2013.

[61] A. Deutsch, L. Sui, V. Vianu, and D. Zhou, "Verification of communicating data-driven web services," in *Proc. of the 25th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2006, pp. 90–99.

[62] W. El Kholy, M. El Menshawy, J. Bentahar, H. Qu, and R. Dssouli, "Verifying multiagent-based web service compositions regulated by commitment protocols," in *Proc. of the IEEE Int. Conf. on Web Services (ICWS)*. IEEE, 2014, pp. 49–56.

[63] M. Ghannoudi and W. Chainbi, "Formal verification for web service composition: A model-checking approach," in *Proc. of the Int. Symp. on Networks, Computers and Communications (ISNCC)*. IEEE, 2015, pp. 1–6.

[64] L. Bordeaux, G. Salaün, D. Berardi, and M. Mecella, "When are two web services compatible?" in *Technologies for E-Services*. Springer, 2004, pp. 15–28.

[65] J. Pathak, S. Basu, and V. Honavar, "On context-specific substitutability of web services," in *Proc. of the IEEE Int. Conf. on Web Services (ICWS)*. IEEE, 2007, pp. 192–199.

[66] H. Dong, F. K. Hussain, and E. Chang, "Semantic web service matchmakers: state of the art and challenges," *Concurrency and Computation: Practice and Experience*, vol. 25(7), pp. 961–988, 2013.

[67] M. Zhang, C. Liu, J. Yu, Z. Zhu, and B. Zhang, "A correlation context-aware approach for composite service selection," *Concurrency and Computation: Practice and Experience*, vol. 25(13), pp. 1909–1927, 2013.

[68] A. Martens, "On compatibility of web services," *Petri Net Newsletter*, vol. 65, no. 12-20, p. 100, 2003.

[69] A. Bracciali, A. Brogi, and C. Canal, "A formal approach to component adaptation," *Journal of Systems and Software*, vol. 74(1), pp. 45–54, 2005.

[70] L. Kuang, Y. Xia, S. Deng, and J. Wu, "Analyzing behavioral substitution of web services based on pi-calculus," in *Proc. of the IEEE Int. Conf. on Web Services (ICWS)*. IEEE, 2010, pp. 441–448.

[71] F. Liu, Y. Shi, L. Zhang, L. Lin, and B. Shi, "Analysis of web services composition and substitution via CCS," in *Data Eng. Issues in E-Commerce and Services*. Springer, 2006, pp. 236–245.

[72] A. Both and W. Zimmermann, "Automatic protocol conformance checking of recursive and parallel bpel systems," in *Proc. of the IEEE 6th European Conf. on Web Services, ECOWS'08*. IEEE, 2008, pp. 81–91.

[73] A. Salah, G. Tremblay, and A. Chami, "Behavioral interface conformance checking for ws-bpel processes," in *Proc. of the Int. MCETECH Conf. on e-Technologies*. IEEE, 2008, pp. 253–257.

[74] N. Guermouche and C. Godart, "Timed model checking based approach for web services analysis," in *Proc. of the IEEE Int. Conf. on Web Services, ICWS*. IEEE, 2009, pp. 213–221.

[75] M. M. Bersani, L. Cavallaro, A. Frigeri, M. Pradella, and M. Rossi, "SMT-based verification of LTL specification with integer constraints and its application to runtime checking of service substitutability," in *Proc. of the 8th Int. Conf. on Software Eng. and Formal Methods (SEFM)*. IEEE, 2010, pp. 244–254.

[76] W. L. Yeung, "A formal and visual modeling approach to choreography based web services composition and conformance verification," *Expert Systems with Applications*, vol. 38(10), pp. 12 772–12 785, 2011.

[77] J. Cámara, G. Salaün, C. Canal, and M. Ouederni, "Interactive specification and verification of behavioral adaptation contracts," *Information and Software Technology*, vol. 54(7), pp. 701–723, 2012.

[78] H. Groefsema, N. van Beest, and M. Aiello, "A formal model for compliance verification of service compositions," *IEEE Trans. on Services Computing*, vol. 10.1109/TSC.2016.2579621, 2016.

[79] K. Huynh, T. Quan, and T. Bui, "Smaller to sharper: efficient web service composition and verification using on-the-fly model checking and logic-based clustering," in *Proc. of the Int. Conf. on Computational Sci. and Its Applications*. Springer, 2016, pp. 453–468.

[80] A. S. Bataineh, J. Bentahar, M. El Menshawy, and R. Dssouli, "Specifying and verifying contract-driven service compositions using commitments and model checking," *Expert Systems with Applications*, vol. 74, pp. 151 – 184, 2016.

[81] Z. Sbaï and R. Guerfel, "Ctl model checking of web services composition based on open workflow nets modeling," *Int. Journal of Service Science, Management, Engineering, and Technology (IJSSMET)*, vol. 7, no. 1, pp. 27–42, 2016.

[82] S. Nakajima, "Model-checking of safety and security aspects in web service flows," in *Proc. of the Int. Conf. on Web Engineering*. Springer, 2004, pp. 488–501.

[83] H. Schlingloff, A. Martens, and K. Schmidt, "Modeling and model checking web services," *Electronic Notes in Theoretical Computer Sci.*, vol. 126, pp. 3–26, 2005.

[84] M. Mongiello and D. Castelluccia, "Modelling and verification of bpel business processes," in *Proc. of the 4th Workshop on Model-Based Development of Computer-Based Systems and 3rd Int. Workshop on Model-Based Methodologies for Pervasive and Embedded Software (MBD-MOMPES'06)*. IEEE, 2006, pp. 5–9.

[85] Z. Xiangpeng, A. Cerone, and P. Krishnan, "Verifying bpel workflows under authorisation constraints," in *Proc. of the Int. Conf. on Business Process Management*. Springer, 2006, pp. 439–444.

[86] Y. Xiaolie and L. Lejian, "Verifying a secure session protocol for web services," in *Proc. of the Int. Conf. on Networks Security, Wireless Communications and Trusted Computing, (NSWCTC'09)*, vol. 2. IEEE, 2009, pp. 301–304.

[87] G. Zhang, M. Rong, Y. He, X. Zhu, and R. Yan, "A refinement checking method of web services composition," in *Proc. of the 5th IEEE Int. Symp. on Service Oriented System Eng. (SOSE)*. IEEE, 2010, pp. 103–106.

[88] Y. Kasse and L. Mokdad, "Quantitative verification for response times in composite web service model," in *Proc. of the IEEE Symp. on Computers and Communications (ISCC)*. IEEE, 2011, pp. 97–102.

[89] H. Gao, H. Miao, and H. Zeng, "Predictive web service monitoring using probabilistic model checking," *Appl. Math*, vol. 7(1L), pp. 139–148, 2013.

[90] J. Lu, Z. Huang, and C. Ke, "Verification of behavior-aware privacy requirements in web services composition," *Journal of Software*, vol. 9(4), pp. 944–951, 2014.

[91] H. Jingjing, Z. Wei, Z. Xing, and Z. Dongfeng, "Web service composition automation based on timed automata," *Applied Mathematics & Inform. Sci.*, vol. 8, no. 4, pp. 2017–2024, Jul. 2014.

[92] S. Rebai, H. H. Kacem, M. Karâa, S. E. Pomares, and A. H. Kacem, "A service-oriented architecture (soa) framework for choreography verification," in *Proc. of the 14th Int. Conf. on Computer and Inform. Sci. (ICIS)*. IEEE, 2015, pp. 642–646.

[93] Z. Rui-Min and L. Xiao-Bin, "Model checking of non-centralized automaton web service with amt bounded constraint," *Int. Journal of Multimedia and Ubiquitous Eng.*, vol. 11, no. 3, pp. 57–66, 2016.

[94] C. Mi, H. Miao, J. Kai, and H. Gao, "Reliability modeling and verification of bpel-based web services composition by probabilistic model checking," in *Proc. of the 14th Int. Conf. on Software Eng. Research, Management and Applications (SERA)*. IEEE, 2016, pp. 149–154.

[95] I. El Kassmi and Z. Jarir, "Security requirements in web service composition: Formalization, integration, and verification," in *Proc. of the 25th Int. Conf. on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*. IEEE, 2016, pp. 179–184.

[96] J. El-Qurna, H. Yahyaoui, and M. Almulla, "A new framework for the verification of service trust behaviors," *Knowledge-Based Systems*, 2017.

[97] Z. Manna and A. Pnueli, *The temporal logic of reactive and concurrent systems: specifications*. springer, 1992, vol. 1.

[98] G. Díaz, M.-E. Cambronero, M. L. Tobarra, V. Valero, and F. Cuartero, "Analysis and verification of time requirements applied to the web services composition," in *Proc. of the Int. Workshop on Web Services and Formal Methods*. Springer, 2006, pp. 178–192.

[99] R. Kazhamiakin, P. Pandya, and M. Pistore, "Timed modelling and analysis in web service compositions," in *Proc. of the 1st Int. Conf. on Availability, Reliability and Security (ARES'06)*. IEEE, 2006, pp. 7 pp.–846.

[100] E. Martinez, M. E. Cambronero, G. Diaz, and V. Valero, "Design and verification of web services compositions," in *Proc. of the 4th Int. Conf. on Internet and Web Applications and Services (ICIW'09)*. IEEE, 2009, pp. 395–400.

[101] M. E. Cambronero, G. Díaz, V. Valero, and E. Martínez, "Validation and verification of web services choreographies by using timed automata," *The Journal of Logic and Algebraic Programming*, vol. 80, no. 1, pp. 25–49, 2011.

[102] G. Rawand, Z. Sbaï, and K. Barkaoui, "Modeling and formal verification framework of web services composition," in *Proc. of the Int. Conf. on Control, Eng. Inform. Technology (CEIT'13)*, vol. 2, 2013, pp. 140–145.

[103] Y. Zhao, H. Xiao, Z. Wang, G. Pu, and T. Su, "The semantics and verification of timed service choreography," *Int. Journal of Computer Mathematics*, vol. 91(3), pp. 384–402, 2014.

[104] S. Nakajima, "Verification of web service flows with model-checking techniques," in *Proc. of the 1st Int. Symp. on Cyber Worlds*. IEEE, 2002, pp. 378–385.

[105] A. Betin-Can, T. Bultan, and X. Fu, "Design for verification for asynchronously communicating web services," in *Proc. of the 14th Int. Conf. on World Wide Web*. ACM, 2005, pp. 750–759.

[106] Z. Xiangpeng, Y. Hongli, and Q. Zongyan, "Towards the formal model and verification of web service choreography description language," in *Proc. of the Int. Workshop on Web Services and Formal Methods*. Springer, 2006, pp. 273–287.

[107] C. Vaz and C. Ferreira, "Towards automated verification of web services," in *Proc. of the IADIS Int. Conf. on WWW/Internet*, 2007.

[108] Y. Hongli, Z. Xiangpeng, C. Chao, and Q. Zongyan, "Model-checking of web services choreography," in *Proc. of the IEEE Int. Symp. on Service-Oriented System Eng. (SOSE'08)*. IEEE, 2008, pp. 79–84.

[109] T. H. Quyet, Q. P. Thi, and D. B. Hoang, "A method of verifying web service composition," in *Proc. of the Symp. on Inform. and Communication Technology*. ACM, 2010, pp. 155–162.

[110] H. Gao, H. Miao, S. Chen, and J. Mei, "Probabilistic timed model checking for atomic web service," in *Proc. of the IEEE World Congr. on Services*. IEEE, 2011, pp. 459–466.

[111] N. Ibrahim and I. Khalil, "Verifying web services compositions using uppaal," in *Proc. of the Int. Conf. on Computer Systems and Industrial Informatics (ICCSII)*. IEEE, 2012, pp. 1–5.

[112] K. T. Huynh, T. T. Quan, and T. H. Bui, "A bitwise-based indexing and heuristic-driven on-the-fly approach for web service composition and verification," *Vietnam Journal of Computer Science*, pp. 1–16, 2016.

[113] G. Diaz and M. Cambronero, "Model checking techniques applied to the design of web services," *CLEI Electronic*, vol. 10(2), 2007.

[114] A. Lomuscio, H. Qu, M. Sergot, and M. Solanki, "Verifying temporal and epistemic properties of web service compositions," in *Proc. of the Int. Conf. on Service-Oriented Computing*. Springer, 2007, pp. 456–461.

[115] S. Wieczorek, V. Kozyura, A. Roth, M. Leuschel, J. Bendisposto, D. Plagge, and I. Schieferdecker, "Applying model checking to generate model-based integration tests from choreography models," in *Testing of Software and Communication Systems*. Springer, 2009, pp. 179–194.

[116] L. Peng, C. Cai, Q. Zongyan, and G. Pu, "Verification of channel passing in choreography with model checking," in *Proc. of the Int. Conf. on Service-Oriented Computing and Applications (SOCA)*. IEEE, 2009, pp. 1–5.

[117] Z. Kang and H. Wang, "Verifying ws-cdl-based web services collaboration by model checking," in *Proc. of the Congr. on Services-I*. IEEE, 2009, pp. 554–561.

[118] Z. Fang, L. Liao, and R. Chen, "Bounded model checking for web service discovery and composition," in *Proc. of the Software Eng. Artificial Intelligence Networking and Parallel/Distributed Computing (SNPD)*. IEEE, 2010, pp. 201–206.

[119] S. Rossi, "Model checking adaptive multilevel service compositions," in *Proc. of the 7th Int. Workshop on Formal Aspects of Component Software, FACS*, 2010, pp. 106–124.

[120] A. Lomuscio, H. Qu, and M. Solanki, "Towards verifying contract regulated service composition," *Autonomous Agents and Multi-Agent Systems*, vol. 24(3), pp. 345–373, 2012.

[121] L. Yongxiang, Y. Xifan, Z. Jie, and L. Bin, "Cloud manufacturing service composition modeling and formal verification based on calculus for orchestration of web service," in *Proc. of the 25th Chinese Control and Decision Conf. (CCDC)*. IEEE, 2013, pp. 2806–2810.

[122] N. Kokash and F. Arbab, "Formal design and verification of long-running transactions with extensible coordination tools," *IEEE Trans. on Services Computing*, vol. 6(2), pp. 186–200, 2013.

[123] S. Bourne, C. Szabo, and Q. Z. Sheng, "Verifying transactional requirements of web service compositions using temporal logic templates," in *Proc. of the 14th Int. Conf. on Web Information Systems Engineering WISE*, 2013, pp. 243–256.

[124] Q. Yu and A. Bouguettaya, "Framework for web service query algebra and optimization," *ACM Trans. on the Web*, vol. 2, no. 1, p. 6, 2008.

[125] R. Liu, C. Hu, and C. Zhao, "Model checking for web service flow based on annotated owl-s," in *Proc. of the 9th ACIS Int. Conf. on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, 2008, pp. 741–746.

[126] A. Airkin, S. Askary, S. Fordin, W. Jekeli, D. Orchard, and K. Riemer, *Web Service Choreography Interface WSCI 1.0*, World Wide Web Consortium Std., 2002.

[127] G. J. Holzmann, *The SPIN model checker: Primer and reference manual*. Addison-Wesley Reading, 2004, vol. 1003.

[128] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "Nusmv 2: An opensource tool for symbolic model checking," in *Proc. of the Computer Aided Verification*. Springer, 2002, pp. 359–364.

[129] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, Y. Zhu *et al.*, "Bounded model checking." *Advances in computers*, vol. 58, no. 11, pp. 117–148, 2003.

[130] D. Beyer, T. A. Henzinger, R. Jhala, and R. Majumdar, "The software model checker blast," *Int. Journal on Software Tools for Technology Transfer*, vol. 9(5-6), pp. 505–525, 2007.

[131] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi, "UPPAAL — a tool suite for automatic verification of real–time systems," in *Proc. of the Workshop on Verification and Control of Hybrid Systems III*. Springer, 1995, pp. 232–243.

[132] L. De Alfaro, M. Kwiatkowska, G. Norman, D. Parker, and R. Segala, "Symbolic model checking of probabilistic processes using mtbdds and the kronecker representation," in *Proc. of the Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2000, pp. 395–410.

[133] A. Lomuscio, H. Qu, and F. Raimondi, "Mcmas: A model checker for the verification of multi-agent systems," in *Proc. of the Int. Conf. on Computer Aided Verification*. Springer, 2009, pp. 682–688.

[134] Y. Liu, J. Sun, and J. S. Dong, "Pat 3: An extensible architecture for building multi-domain model checkers," in *Proc. of the 22nd Int. Symp. on Software Reliability Eng. (ISSRE)*. IEEE, 2011, pp. 190–199.

[135] K. Schmidt, "Lola a low level analyser," in *International Conference on Application and Theory of Petri Nets*. Springer, 2000, pp. 465–474.

[136] H. Wang, C. Wang, and Y. Liu, "A logic-based approach to web services composition and verification," in *Proc. of the World Conf. on Services-II, SERVICES-2'09*. IEEE, 2009, pp. 103–110.

[137] H. Wang, L. Li, C. Wang, Z. Kang, D. Liu, J. Wu, and A. Bouguettaya, "Logic-based verification for web services composition with TLA," in *Proc. of the Int. Conf. on Service-Oriented Computing and Applications (SOCA)*. IEEE, 2009, pp. 1–8.

[138] H. Huang, W.-T. Tsai, and R. Paul, "Automated model checking and testing for composite web services," in *Proc. of the 8th IEEE Int. Symp. on Object-Oriented Real-Time Distributed Computing (ISORC'05)*, 2005, pp. 300–307.

Gopal N. Rai received his B.Sc. and M.C.A. degree from University of Allahabad, Allahabad, India, in 2007 and 2011, respectively. He is currently working towards the Ph.D. degree at the University of Hyderabad and the Institute for Development and Research in Banking Technology (IDRBT), Hyderabad. His research interests include Software Verification, Web services, and formal methods.

G R Gangadharan is an associate professor at the Institute for Development and Research in Banking Technology, Hyderabad, India. His research interests focus on the interface between technological and business perspectives. Gangadharan received his PhD in information and communication technology from the University of Trento, Italy, and the European University Association. He is a senior member of IEEE and ACM. Contact him at geeyaar@gmail.com.