Towards a performance analysis of composite web services using Petri nets

Nassima Bernine*

Department of Operational Research,
Faculty of Exact Sciences,
Research Unit (LaMOS) (Modeling and Optimization of Systems),
Bejaia University,
06000 Bejaia, Algeria
Email: nassima.bernine@gmail.com
*Corresponding author

Hassina Nacer

MOVEP Laboratory,
Department of Computer Science,
Science and Technology USTHB' University,
Algiers, Algeria
Email: sino_nacer@yahoo.fr

Djamil Aissani

Research Unit (LaMOS) (Modeling and Optimization of Systems),
Bejaia University,
06000 Bejaia, Algeria
Email: lamos_bejaia@hotmail.com

Hassane Alla

GIPSA Laboratory,
Department of Automatic,
Grenoble University, France
Email: Hassane.Alla@gipsa-lab.grenoble-inp.fr

Abstract: The proliferation of web services-based applications, collaboration, interoperability between companies, and more generally demands of web services over the internet are major challenges in the design of performance evaluation in web services discovery and composition. It is required for the web services providers to analyse the performance related to the satisfaction of clients requests. This paper presents an analytical model based on Petri nets, for evaluating the performance of a web services system where requests and web services follow an exponential server. The arrival process of the users requests and web services follows a Poisson distribution. When the web services finish the service, they join the queue. This model follows a Poisson distribution, it is based on Petri nets. We use an analytical method for solving the model. Furthermore, we compute the system's response time,

and the average number of clients in the system in terms of the arrival rate of clients requests. We find the limit number of clients in the system from which it begins to be saturated.

Keywords: performances analysis; requests satisfaction; composite web services; Petri nets model; analytical model; Poisson distribution

Reference to this paper should be made as follows: Bernine, N., Nacer, H., Aissani, D. and Alla, H. (2020) 'Towards a performance analysis of composite web services using Petri nets', *Int. J. Mathematics in Operational Research*, Vol. 17, No. 4, pp.467–491.

Biographical notes: Nassima Bernine earned her Magister in Applied Mathematics at the Department of Operations Research, Faculty of Exact Sciences at the University of Bejaia, Algeria. She is currently preparing her PhD thesis on Performance Evaluation of Web Services in the Bejaia University, Algeria. She is a permanent researcher at the Research Unit LaMOS. Her research interests include Petri nets, queuing theory, web services system and their performances evaluation.

Hassina Nacer received her PhD from the Bejaia University in 2010 in Computer Science. From 2001 to 2014, she was a teacher-researcher in the Bejaia University. In 2015, she received her Habilitation to Direct Research (HDR) in USTHB. She has taught in several universities (Bejaia, USTHB, EMP, Alger1). Her research interests include web services technology, web semantic, knowledge base systems, distributed and cooperative computing, web services security, and cloud computing. She published various papers in international journals.

Djamil Aissani is a Full Professor of Mathematics at the Department of Operations Research at the University of Bejaia, Algeria. He started his career at the University of Constantine in 1978. He received his PhD in 1983 from the Kiev State University, Soviet Union. He is at the University of Bejaia since its opening in 1983/1984. He is also the Head of the Faculty of Science and Engineering from 1999 to 2000, the Director of the Research Unit LaMOS, the Scientific Head of the Computer Science Doctorate School ReSyD from 2003 to 2011, and he has taught in many Universities (USTHB Algiers, Annaba, Rouen, Dijon, ENITA, EHESS Paris, CNAM Paris, etc.). He has published many papers on Markov chains, queueing systems, reliability theory, performance evaluation and their applications in electrical, telecommunication networks and computer systems.

Hassane Alla is a Professor at the Grenoble University and a researcher in the Gipsa-Lab. His research is mainly concerned with tools derived from Petri nets, used for the performance evaluation and for the control synthesis of discrete event systems. He is an author or co-author of 150 publications. One of its main publications is a book on continuous and hybrid Petri nets which has been published in English and in French.

1 Introduction

In order to survive the massive competition created by the new online e-economy and collaborative working, several companies put their skills on the internet as web services.

Web services are based on extensible markup language (XML).[1] They are usually described with standards like (UDDI, SOAP, WSDL). Universal description, discovery and integration (UDDI) is a virtual registry that exposes information about web services. Simple object access protocol (SOAP) is a protocol used to exchange structured information in a decentralised and a distributed environment. It uses XML to define an extensible framework of messages, which provides a constructed message that can be exchanged through a variety of underlying protocols. The SOAP protocol is independent of any particular programming model and from any specific semantics of implementation. Web services description language (WSDL) provides a model and an XML format for describing the web services. It separates the description of the abstract functionality, offered by a service, from the concrete details of a service description such as 'how?' and 'where?'. WSDL describes only the syntactic interface of web services.

The web services technology gives a new dimension to the inter-operability; but at a higher level, the basic infrastructure is not sufficient to guarantee a great capacity of cooperation and to satisfy the client's needs (Nacer et al., 2009). This means that a web service has a specific task to perform and may depend on other web services, hence being composite. The web services composition is useful as soon as a client's request cannot be satisfied by a simple existing service but by a cooperation and a combination between several simple web services (Nacer and Aissani, 2014).

The research axis of web service discovery and composition (Niu et al., 2019; Weitao and Wei, 2018), giving a virtual web service, has recently become very popular. The requirement of requests satisfaction presents a challenge to web services providers given the large scale, the distributed and heterogenous nature of the infrastructure supporting such services. Several solutions have been proposed to address the above challenge. However, one of the most relevant research issues is a modelling technique to ensure a high analysis and performance for the clients requests asking a virtual web service (Yugen and Yonggang, 2010; Nacer et al., 2017), which can be simple or composite.

The efficient service compositor considers several candidate services with the similar functions and dissimilar quality of service (QoS) limitations. For selecting suitable services from a service pool, addressing service composition restrictions, determining the important QoS parameters, understanding the dynamic features of the problem, and having fast changes in the properties of the services and network are some significant issues that must be addressed to ensure the service user's satisfaction. Also, there are some QoS factors in each cloud service for service evaluating (Naseri and Navimipour, 2019). QoS attributes related to web services are (Mohan Reddy et al., 2013):

- Scalability: The throughput improvement rate in a given time interval.
- Performance: A measure of the speed in completing a service request. It is measured by:
  - a Latency: The time taken to start servicing a requested service.
  - b Throughput: A service request processing rate.

c Response time: The time taken between the end of a service or a demand of a service and the beginning of a response.

- Capacity: The number of parallel requests that the service allows.
- Availability: The percentage of service operating time.
- Reliability: The time for continuity of expected service and for transition to accurate state (see for example Weitao and Wei, 2018).
- Accuracy: The service error rate in a specified time interval (Jun and Jian, 2019).
- Robustness: The service flexibility level to inappropriate access and invocation services.
- Stability: The change rate of service.
- Cost: The total cost to use the service (Matteo et al., 2019).
- Security: The specification of the methodologies to protect data.
- Reliable: messaging Determines if the service offers mechanisms to guarantee reliable message delivery.
- Integrity: Determining the transactional properties support to the services.
- Interoperability: Checking if the service is acquiescent with interoperability profiles (Mecheri et al., 2019).

The motivation of the proposed model is more about solving the problem of saturation of the web services system. Indeed, in the last few decades, the vision of web services moved to the composition of web services. Thus, our main contribution is to model a web services system with Petri nets, taking into account two types of arrivals: web services and clients requests arrivals. Then, we analyse and calculate the performance of the proposed model. Thanks to this analysis, we gain analytical results as a number of web services in the system and the influence of the arrival rate of clients requests on the system. The paper is organised as follows. Section 2 presents problem statement. Section 3 presents basic concepts on Petri nets and web services. Section 4 resumes the literature review on performance evaluation in web services, Section 5 describes the modelling approach for web services and presents sensitivity analysis results. In Section 6, we present position of our results. Section 7 concludes the paper.

2 Problem statement

Despite the success of web services technology, the latter is currently facing a number of limitations. While the development, the publication and the invocation of simple web services are guided by web services standards (XML, WSDL, SOAP, UDDI).

In the literature, we find works where the authors improve these standards (Zhong and Qi, 2006; Sato and Trivedi, 2007) improve BPEL (Boutrous Saab et al., 2006; Rosenber et al., 2010) enhance the SOAP standard in order to improve the composition of the web services, and the QoS's. The selection of web services is improved in Wang et al. (2006). Our objective consists on modelling with stochastic Petri nets. This approach allowed us to have an analytical model of the web services system

that takes into account two different types of arrivals: client request and web services. The performances of this model were calculated using an analytical method. The limit number of client request and web services in the system to saturate it, is also obtained. With the study of performance evaluation of the web services, we obtains the influence of client request arrivals on the web services system, which guarantees its unsaturation.

3 Concepts of bases

3.1 Stochastic Petri nets

Petri nets are graphical and mathematical tools, making them possible to model and to check the dynamic behaviour of the systems with discrete events. They are directed bipartite graph with two node types called places and transitions. The nodes are connected via directed arcs. Connections between two nodes of the same type are not allowed. Places are represented by circles and transitions by rectangles (Comet, 2014).

Definition: A generalised stochastic Petri nets (GSPN) is a six-tuple (P, T, F, W, M_0, λ). $P = \{p_1, p_2, ..., p_k\}$ is a finite set of places. T is a finite set of transitions partitioned into two subsets: T_I (immediate) and T_D (timed) transitions, where transition $t \in T_D$ are associated with rate λ. $F \subseteq (P \times T) \cup (T \times P)$ a set of arcs. $M_0 = \{m_{01}, m_{02}, ..., m_{0k}\}$ is an initial marking. $W : T \to R$ is a function defined on the set of transitions. Timed transitions are associated with priority zero, whereas all other priority levels are reserved for immediate transitions. The immediate transitions are drawn as thin bars, while the timed transitions are drawn as rectangles (Zhong and Qi, 2006).

3.2 Web services discovery and composition: technology of web services

As commonly assumed in the literature, a web service is a component of distributed applications, which provide data to other applications by using the communication infrastructure offered by the web. It respects some properties such as autonomous object component, slightly coupled, self-describing, synchronous and asynchronous. To understand the web services composition, we combine several web services to create a composite web service. According to Gardarin (2002), web services composition is a technique which assembles web services in order to achieve a particular goal, via primitives of control (test, treatment of exception, ...) and exchange (sending and reception of messages).

According to Fensel et al. (2002), composition is a process which functions in an intelligent way in order to discover services automatically, and allows them to be combined in a more complex way.

We can say that web services composition is the use of relevant web services offered by various providers in order to obtain a composite service able to satisfy a user's request which cannot be satisfied by a simple available web service.

An automatic and dynamic web services composition is a highly complex task, because the proposed standards (XML, WSDL, UDDI, SOAP) of web services technology do not answer the problems of web services discovery and composition by

a software agent. In addition, the semantic annotations of web services and requests are not yet mature.

3.2.1 Standards of web services

3.2.1.1 XML (Nacer and Aissani, 2014)

XML is a standard of W3C is a universal model of data representation and exchange. It is a simple format text, flexible and also independent of any manufacturer. Adding to this, it gives structure to documents and data. It is extracted from SGML language and it benefits from experiences of hyper text mark-up language (HTML's) use. Further more, XML offers portable and structured data on heterogeneous structure and programming languages. XML brings the following criteria to XML web services architecture:

- Extensibility: A system can function correctly without losing its main properties during an update.
- Neutrality: The required constraints of an application are limited.
- Structure: XML represents both document structure and content, offers increased control of information granularity through transformation and query languages.
- Interoperability: Communication and data exchange between heterogeneous systems are possible.

3.2.1.2 SOAP: simple object access protocol (Nacer and Aissani, 2014)

The SOAP protocol is an exchange message's process in heterogeneous environments for application-to-application communication based on XML and on standard protocol HTTP. SOAP a standard of W3C, defines a set of rules to structure dialogues remote procedure call (RPC) to exchange data. It ensures interoperability between components independent of transport mechanisms, operating systems and programming languages. SOAP is a flexible protocol to connect distributed systems. The purpose of this protocol is to facilitate the access to software services to any user through the internet.

3.2.1.3 WSDL: web services description language (Nacer and Aissani, 2014)

WSDL is a formal language of web services description according to the standard XML. A WSDL file describes the functionality (methods, parameters) and the localisation of a web service (URI, port, and protocol of invocation). According to W3C (http://www.w3.org/TR/wsdl), WSDL separates the description of abstract functionalities offered by a service from concrete details of service description. As in programming languages, a type signature defines the inputs and outputs for a function. It means that WSDL can be seen as a traditional function, subroutine or method.

3.2.1.4 UDDI: universal description, discovery and integration (Nacer and Aissani, 2014)

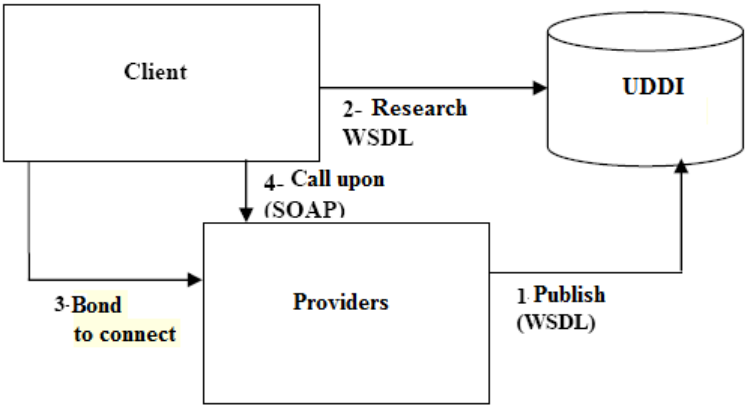
The registry of web services 'UDDI' is a virtual data base of existing XML web services. It is similar to a CORBA trader and can be considered as a DNS service for business applications. On one hand, it allows providers of services to record XML web

services under a standardised format and on the other hand, it concentrates on discovery process of XML web services satisfying services' needs in SOA. UDDI becomes an intermediate standard between providers and clients through the internet and it is a recommendation of W3C. The UDDI project is an initiative of industry which tries to create an independent platform, to describe services, to discover businesses and to integrate services. It means that UDDI provides a universal registry for business to provide service listings (web service description).

3.2.2 Processing of web services

In the web service composition approach implementing SOA architecture, the three entities (provider, client, UDDI) see Figure 1 interact (Nacer and Aissani, 2014).

Figure 1 Life-cycle process of the system



3.2.2.1 Publication

The provider publishes simple web services in UDDI.

3.2.2.2 Search

The client searches the web service which satisfies his request.

3.2.2.3 Discovery

The UDDI is explored in order to discover the simples web services that meet the client's request.

3.2.2.4 Composition

It allows to find the web service requested, composite web services when simple web services are not found.

Web services composition includes two processes (Nacer and Aissani, 2014):

- Discovery of user's goals: A process which translates the goals of a user into precis and formal goals.

- Composition of discovered services: A process which meets the awaited needs for a user.

The response to a request's user may be one of the eight following cases;

- Case 1: The user's goals are completely covered by a single web service (the user's needs and the announced possibilities of the single service are perfectly matched).
- Case 2: The user's goals are completely covered by several web services (the user's needs and the announced possibilities of services composition are perfectly matched).
- Case 3: The user's goals can be completely covered by a single web service. However,the user can receive objects which are not suitable for him (opposite subsumption).
- Case 4: The user's goals can be completely covered by several web services. However, the user can receive services which are not suitable for him (opposite subsumption).
- Case 5: The user's goals can not be completely covered by a single web service (the user's needs and the announced possibilities of the single service are partially equivalent (subsumption). Nevertheless, the discovered service does not provide any non-suitable object for the user.
- Case 6: The user's goals can not be completely covered by several web services (the user's needs and the announced possibilities of several services are partially matched). Nevertheless, the discovered services do not provide any non-suitable object for the user (subsumption).
- Case 7: The user's goals are completely covered by a single web service. Nevertheless, the discovered service provides non-suitable object for the user.
- Case 8: The user's goals are disjoined of the announced possibilities of all existing web services.

In our work, we focus on the composition of web services.

4 Related work

During the last few years, the problem of web services performance evaluation has received a lot of attention by many researchers, in order to improve the clients satisfaction.

In Chandrasekaran et al. (2003), proposed a service composition an execution tool (SCET), where a web process presented as a graph using the process designer's source nodes, sink nodes, activity nodes, data links and control links. SCET is integrated with the JSIM simulator, enabling users to simulate a process and get statistical performance estimates. In Zhong and Qi (2006), modelised and transformed the BPEL into a stochastic Petri nets model, using a reliability prediction technique [algorithm of stochastic Petri net Package (SPNP)] that takes into account the structure of BPEL, and

the concurrent nature of service composition. In Boutrous Saab et al. (2006), proposed a transformation of the requests into SOAP message for the composition of the web services. They calculated the average time of response in terms of the arrival rate of requests. In Sato and Trivedi (2007), developed an approach for service composition by combining existing services using a high-level language (BPEL) and the overall reliability of composition web services (Zheng et al., 2017), based on the simulation of BPEL processes. For evaluating this approach, they defined a BPEL process for an example and run it on IBM WebSphere Process Server (v6.0). In Haddad et al. (2013), decomposed the requests in sub-queries to different elementary web services, and then were merged into a final result. They assumed that an elementary web service can be invoked with a constant probability. They broke up the requests into 'n' elementary services, where each elementary services will call upon 'n' web services. They calculated the response time of clients in terms of service rate. In Mokdad et al. (2015), under the same hypotheses as in Haddad et al. (2013), use another method (stochstic automata networks) to calculate the average response time, by taking into account the number of web services with respect to whether they are: constant or variable. In Kumar (2015), calculated the response time of clients, this performance was monitored using a monitoring tool (pingdom). In Chattopadhyay and Banerjee (2017), proposed a new approach for efficient service composition based on abstraction refinement. Instead of considering individual services during composition, they proposed several abstractions to form service groups and the composition was done on these abstract services, using an algorithm for an abstraction procedure implemented in Java. All experiments were performed on an ubuntu, Linux system, and they calculated the number of abstract and average composition time. In Deng et al. (2016), proposed a reliability calculation method for web service compositions, which uses fuzzy reasoning coloured Petri net (FRCPN), to verify the web service compositions.

In Chemaa et al. (2012), show how simple existing web services can be composed (Pinto et al., 2017), in order to create a composite service, which offers new features. In this context, they propose an expressive object-oriented Petri net based algebra that succeeds in the complex composition of web services. In Wang et al. (2006), proposed a QoS-aware web services selection (Boustil et al., 2016) model based on linear programming fuzzy technology (Reza et al., 2018), to identify their dissimilarity on the solutions of the applications, and to help customers to choose the appropriate web services to those needs and preferences. The authors proposed a selection algorithm based on the model linear programming techniques for multidimensional analysis of preferences (LINMAP). To find the optimal QoS, the authors calculated the consistency and measurement of inconsistency customers and the square distance weighing the ideal solution. Then, in Klein et al. (2010), considered the possible compositions of web services, and find all the services that optimise (Abdulqader and Nasruddin, 2018; Ansorena, 2019) some attributed QoS under given constraints QoS that are NP-hard. As in reality, there are many web services running at the same time, therefore, the authors modified the problem with considering a repeated of the same web service. Sam et al. (2018) modelled the modified problem with linear programming, and calculated the number of web services task. In Rosenber et al. (2010), have presented an optimisation approach for the composition of web services for systems service-oriented architecture (SOA) on a large scale, which is the subject of the constraints of QoS (Ding et al., 2016). In particular, the authors influence on the composition model for the flexible specification of QoS constraints by employing hierarchical constraints. In Csizmar

(2001), considered a system of web server with eager users. The system is modelled by a simple queue with poissonian arrivals. The authors were interested in the revenue that the server has to win, and they have developed expressions for the expected revenue. In Hoecke et al. (2005), presented an analytical model to evaluate the impact of using web services as middle ware on the overall system performance. Specifically, layered queueing networks are used to model the web service middle ware layer. A simulation program is implemented in C++, to calculate: the average service time. In Singh and Pattanaik (2013), propose a model to quantify availability and reliability of atomic services using Markov chain model and Weibull analysis respectively. Markov chain models are used to study systems that could be represented as discrete states. In a realistic services scenario, failure rate can be increasing, decreasing, or constant. To accommodate all these cases Weibull analysis has been considered. Moreover, CPN-based simulation experiments are reported, demonstrating the effectiveness and the benefits of the proposed methodology. In Reddy et al. (2011), have developed a model of web services using unified modelling language (UML), use case diagram and sequence diagram, and deployment diagram. They obtain the performance metrics by simulating the web services model using a simulation tool simulation of multi-tier queuing architecture (SMTQA). They calculated: the average response time, the average waiting time, the average service time, probability of idle server and the probability of dropping of sessions.

In Garima et al. (2017), present the composed services with different performance parameters. In order to resolve the composed services, combining of different approaches were used. The selection of the services depends on QoS-parameters, which have done by evaluating its performance metrics using sub-optimal and heuristic approaches (Bjork and Mezei, 2016; Haddadi et al., 2018), and implemented it in Java or C sharp language on Ubuntu Linux system or in Microsoft Visual Studio. They calculate: response time and turn around time.

The references codification in the present section is used to refer them in Table 1, Figures 2, 3 and 4.

4.1 Comparative study

Table 1 illustrates a comparative study between the above works about performance evaluation of web services regarding the following criteria:

- Model: It identifies the used model methods.
- Solution: It identifies a solution method, we have analytical method, simulation and prototype.
- Performances evaluation: It describes the metric of performances evaluation, we have a metric of QoS attributes and other performances.
- Web services type: It specifies the web services type, simple or composite web services.

Mark X: means that the authors took into account the characteristic.

In summary, we believe that the most of the existing approaches try to look for a suitable composite web services. These approaches do not handle web services and client requests arrival.

Table 1 A comparative study of works on the evaluation of performance of the web services

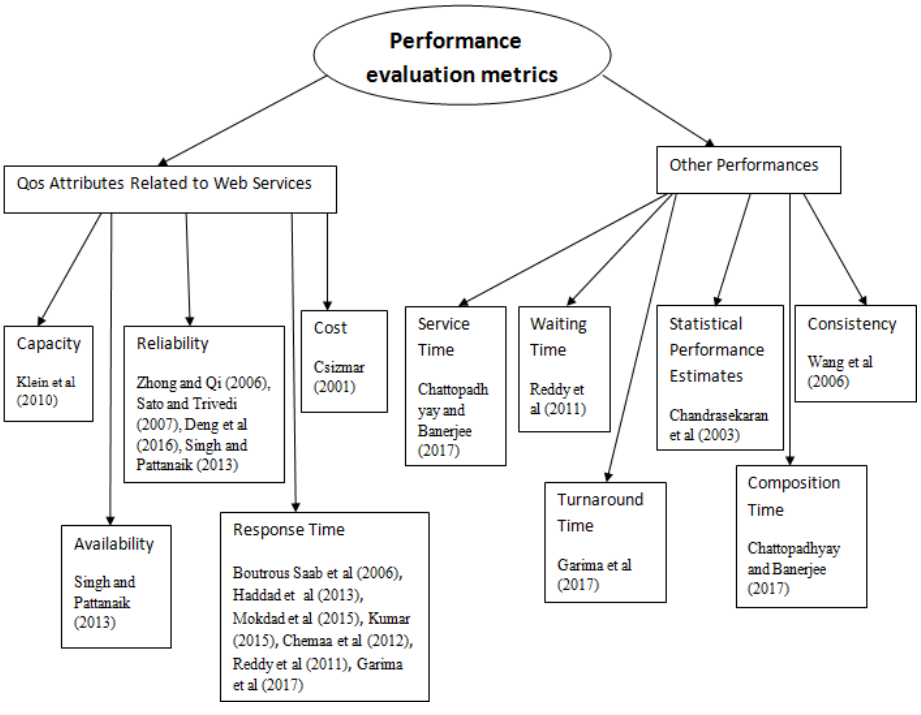
		Chandrasekaran et al. (2003)	*Zhong and Qi (2006)*	*Boutrous Saab et al. (2006)*	*Sato and Trivedi (2007)*	*Haddad et al. (2013)*	*Mokdad et al. (2015)*	*Kumar (2015)*	*Chattopadhyay and Banerjee (2017)*	*Deng et al. (2016)*	*Chemaa et al. (2012)*	*Wang et al. (2006)*	*Klein et al. (2010)*	*Rosenber et al. (2010)*	*Csizmar (2001)*	*Hoecke et al. (2005)*	*Singh and Pattanaik (2013)*	*Reddy et al. (2011)*	*Garim a et al. (2017)*
Model	Queue networks					x									x	x			
	Markov chain																x		
	Linear programming												x						
	Heuristic approach																		x
	Optimisation approach													x					x
	Stochastic automata networks						x												
Solution	Analytical solution					x	x						x	x	x				
	Simulation								x		x	x				x	x	x	x
	Prototype																		
Qos attributs	Capacity												x						
	Availability																x		
	Reliability		x		x					x							x		
	Response time			x		x	x	x			x							x	x
	Cost														x				
Other performances	Consistency											x							
	Statistical performance estimates	x																	
	Composition time								x										
	Turnaround time																	x	
	Waiting time																	x	
	Service time													x		x		x	
Web services	Simple											x			x	x	x	x	
	Composite	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

4.2 Performance evaluation metrics

We summarise a performance evaluation metric calculated in the above works about performance evaluation of web services. This study shows that in the majority of works, the authors calculate a QoS attributes related to web services like response time. However, other authors computed several performances.

Figure 2 shows our summarise regarding performance evaluation metrics calculated in the above works.

Figure 2 Performance evaluation metrics



4.3 Classification of methods

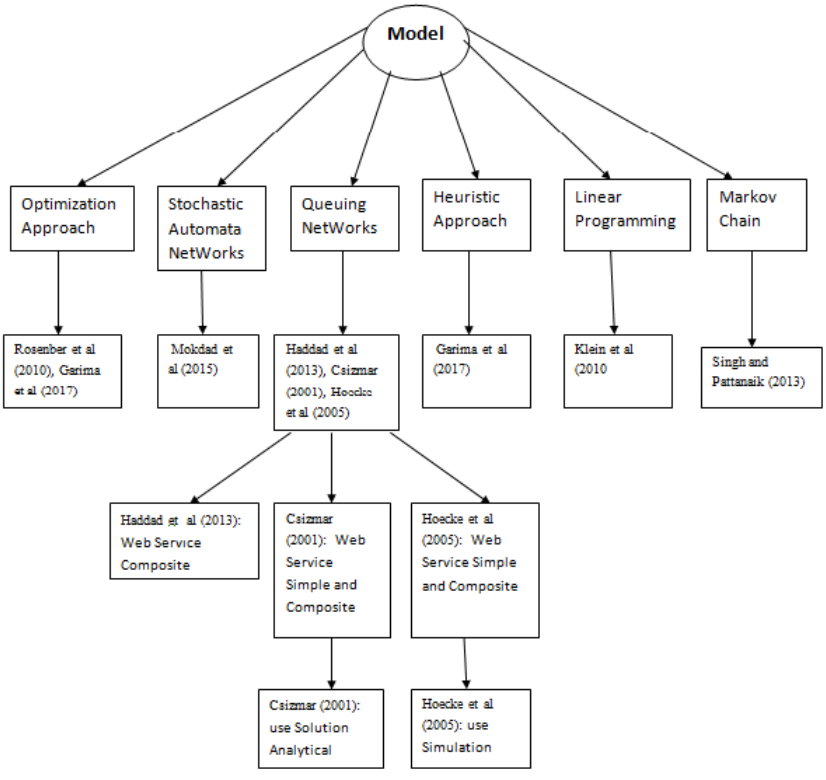
We presented a classification of methods used for modelling the system of web services in the above works about performance evaluation of web services. This study shows that the authors used a different methods to have a model for web services system.

Figure 3 shows our proposed classification of the most known approaches of web services analysis in the above works.

4.4 Solution

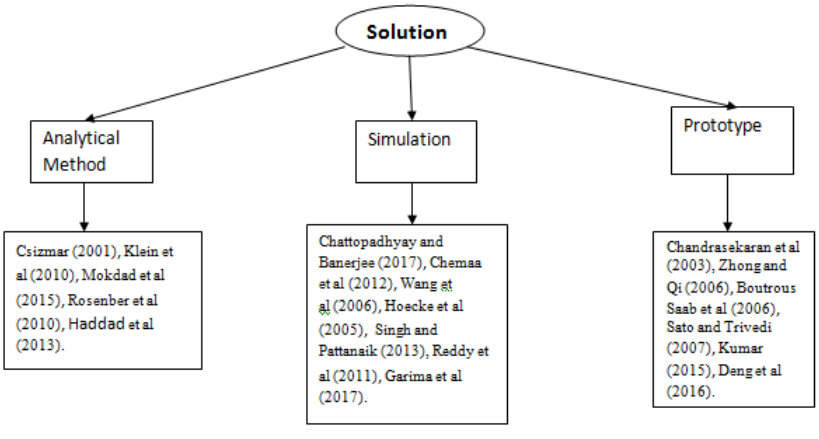
We summarise a solution methods used for calculate a performances of the system of web services model in the above works about performance evaluation of web services. This study shows that the authors used different methods to obtain a system performances of web services model.

Figure 3 The most known models in performance analysis of web services



Performances evaluation methods used in the literature are resumed in Figure 4.

Figure 4 Solution used in the literature



The goal of this paper is to evaluate clients requests and web services taking into account explicitly:

- the architecture characteristics, considering the number of servers
- the traffic model, describing the requests traffic characteristics
- the traffic model, describing the web services traffic characteristics.

Analytical performability models is developed to analyse the impact of the above aspects on the requests and web services. We obtained equations, for which numerical studies were conducted to provide sensitivity analysis with respect to the architecture and traffic characteristics.

Based on the above observations, we consider a Petri nets model with synchronisation of two queues, where the first represents clients requests and the second represents web service requests.

5 The proposed approach

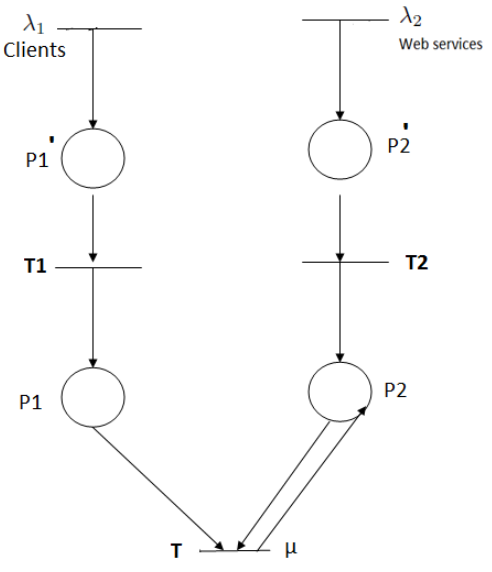
5.1 Modelling requests satisfaction in composite web services

We can compose a web services and we obtain a composite web service (Nacer and Aissani, 2014; Yugen and Yonggang, 2010; Nacer et al., 2017). The latter is not a property of any provider since the several component web services are produced by different autonomous providers (Nacer and Aissani, 2014; Nacer et al., 2017); The evaluation of the requests satisfaction is carried out taking into account the architecture of the system and traffic characteristics. The proposed model describes the system of web services, were the clients requests and web services arrival are taken into account to evaluate these performances. We consider that the system is composed of two stations: the first represents a client request and the second a web services request, with one exponential server. Arrivals to the network follows the Poisson distribution, and the arrival rates are λ_i, $i \in [1, 2]$ and are independent of the failure process. When the clients requests and web services come to the system, they are placed in the place $P_1^{'}$, $P_2^{'}$ respectively. We select n clients to come to the place P_1 and m web services to come to the place P_2, after crossing the transitions T_1 and T_2 respectively. Then, they crosse the transition T with a rate μ to serve the client request. When the service (discovery and composition) is completed, the web services requests rejoin their station. Service rate (discovery and composition) is μ. Figure 5 presents a Petri nets model of a web services system. The graphical representation of the web service model is depicted in Figure 5.

The meanings of places $(P_1^{'}, P_2^{'}, P_1, P_2)$ and transitions (T_1, T_2, T) in the model are:

- $P_1^{'}$: Clients queue, unlimited place.
- $P_2^{'}$: Web services queue, unlimited place.
- P_1: Clients queue, place with limited capacity.
- P_2: Web services queue, place with limited capacity.
- T_1: Election of a clients requests.
- T_2: Election of a web services.
- T: Clients service.

Figure 5 Petri nets model



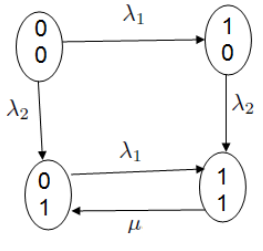
5.2 Mathematical formulation

During the evolution of the system, the marking is likely to be modified. The initial marking, M_0, of a Petri nets corresponds to the initial distribution of the chips in each Petri nets places, which specifies the initial state of the system. We note $M(p)$ the number of chips contained in the place P for the marking M. Then, the initial marking: $(M_0 = [M_0(P1), M_0(P2)] = (0, 0)$ We note by m the number of web services, and by n the number of clients in the system.

For $n = m = 1$.

The Markov chain associated with the Petri nets model is given by Figure 6.

Figure 6 Markov chain of the Petri nets model for $n = m = 1$



From the Markov chain, we will have the instant matrix.

The instant matrix of the system is given by $P^{'}$:

$$P^{'} = \begin{pmatrix} -(\lambda_1 + \lambda_2) & \lambda_1 & \lambda_2 & 0 \\ 0 & -\lambda_2 & 0 & \lambda_2 \\ 0 & 0 & -\lambda_1 & \lambda_1 \\ 0 & 0 & \mu & -\mu \end{pmatrix} \tag{1}$$

The performance evaluation is based on the calculation of the stationary probabilities by solving the system:

$$\begin{cases} \pi.P^{'} = 0 \\ \sum_{i=1}^{4} \pi_i = 1 \end{cases} \tag{2}$$

$$\begin{cases} -(\lambda_1 + \lambda_2)\pi_1 = 0 \\ \lambda_1\pi_1 - \lambda_2\pi_2 = 0 \\ -\lambda_1\pi_3 + \mu\pi_4 = 0 \\ \lambda_1\pi_3 - \mu\pi_4 = 0 \\ \sum_{i=1}^{4} \pi_i = 1 \end{cases} \tag{3}$$

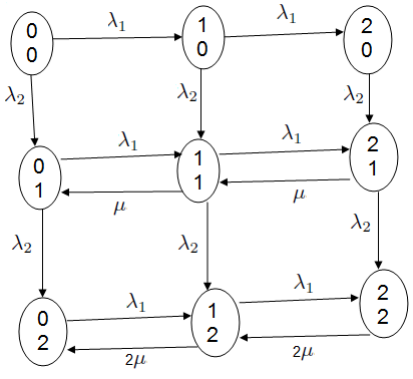
Thus, we obtain The stationary probabilities:

$$\begin{cases} \pi_1 = 0 \\ \pi_2 = 0 \\ \pi_3 = \frac{\mu}{\lambda_1+\mu} \\ \pi_4 = \frac{\lambda_1}{\lambda_1+\mu} \end{cases} \tag{4}$$

For $n = m = 2$.

The Markov chain associated with the Petri nets model is given by Figure 7.

Figure 7 Markov chain of the Petri nets model for $n = m = 2$



From the Markov chain, we will have the instant matrix. The instant matrix of the system is given by $P^{'}$:

$$P^{'} = \begin{pmatrix} -(\lambda_1+\lambda_2) & \lambda_1 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & -(\lambda_1+\lambda_2) & \lambda_1 & 0 & \lambda_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\lambda_2 & 0 & 0 & \lambda_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & -(\lambda_1+\lambda_2) & \lambda_1 & 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & 0 & \mu & -(\lambda_1+\lambda_2+\mu) & \lambda_1 & 0 & \lambda_2 & 0 \\ 0 & 0 & 0 & 0 & \mu & -(\lambda_2+\mu) & 0 & 0 & \lambda_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\lambda_1 & \lambda_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2\mu & -(\lambda_1+2\mu) & \lambda_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\mu & -2\mu \end{pmatrix} \quad (5)$$

The performance evaluation is based on the calculation of the stationary probabilities by solving the system:

$$\begin{cases} \pi.P^{'} = 0 \\ \sum_{i=1}^{9} \pi_i = 1 \end{cases} \quad (6)$$

$$\begin{cases} -(\lambda_1+\lambda_2)\pi_1 = 0 \\ \lambda_1\pi_1 - (\lambda_1+\lambda_2)\pi_2 = 0 \\ \lambda_1\pi_2 - \lambda_2\pi_3 = 0 \\ \lambda_2\pi_1 - (\lambda_1+\lambda_2)\pi_4 = 0 \\ \lambda_2\pi_2 + \lambda_1\pi_4 - (\lambda_1+\lambda_2+\mu)\pi_5 = 0 \\ \lambda_2\pi_3 + \lambda_1\pi_5 - (\lambda_2+\mu)\pi_6 = 0 \\ \lambda_2\pi_4 - \lambda_1\pi_7 + 2\mu\pi_8 = 0 \\ \lambda_2\pi_5 + \lambda_1\pi_7 - (\lambda_1+2\mu)\pi_8 + 2\mu\pi_9 = 0 \\ \lambda_2\pi_6 + \lambda_1\pi_8 - 2\mu\pi_9 = 0 \\ \sum_{i=1}^{9} \pi_i = 1 \end{cases} \quad (7)$$

Thus, we obtain the stationary probabilities:

$$\begin{cases} \pi_1 = 0 \\ \pi_2 = 0 \\ \pi_3 = 0 \\ \pi_4 = 0 \\ \pi_5 = 0 \\ \pi_6 = 0 \\ \pi_7 = \frac{4\mu^2}{\lambda_1^2+4\mu^2+2\lambda_1\mu} \\ \pi_8 = \frac{2\mu\lambda_1}{\lambda_1^2+4\mu^2+2\lambda_1\mu} \\ \pi_9 = \frac{\lambda_1^2}{\lambda_1^2+4\mu^2+2\lambda_1\mu} \end{cases} \quad (8)$$

For $n = m = 3$.

The Markov chain associated with the Petri nets model is given by Figure 8.

From the Markov chain, we will have the instant matrix. The instant matrix of the system is given by $P^{'}$:

$$P' = \begin{pmatrix}
-(\lambda_1+\lambda_2) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -(\lambda_1+\lambda_2) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -(\lambda_1+\lambda_2) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -\lambda_2 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -(\lambda_1+\lambda_2) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \mu & -(\lambda_1+\lambda_2+\mu) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mu & -(\lambda_1+\lambda_2+\mu) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \mu & -(\lambda_2+\mu) & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -(\lambda_1+\lambda_2) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\mu & -(\lambda_1+\lambda_2+2\mu) & \lambda_1 & 0 & 0 & \lambda_2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\mu & -(\lambda_1+\lambda_2+\mu) & \lambda_1 & 0 & 0 & \lambda_2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\mu & -(\lambda_2+2\mu) & 0 & 0 & 0 & \lambda_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda_1 & \lambda_1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3\mu & -(\lambda_1+3\mu) & \lambda_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3\mu & -(\lambda_1+3\mu) & \lambda_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3\mu & -3\mu
\end{pmatrix}$$

The performance evaluation is based on the calculation of the stationary probabilities by solving the system:

$$\begin{cases} \pi.P^{'} = 0 \\ \sum_{i=1}^{16} \pi_i = 1 \end{cases} \tag{9}$$

$$\begin{cases}
-(\lambda_1+\lambda_2)\pi_1 = 0 \\
\lambda_2\pi_1 - (\lambda_1+\lambda_2)\pi_2 = 0 \\
\lambda_2\pi_2 - (\lambda_1+\lambda_2)\pi_2 = 0 \\
\lambda_1\pi_3 - \lambda_2\pi_4 = 0 \\
\lambda_1\pi_1 - (\lambda_1+\lambda_2)\pi_5 + \mu\pi_6 = 0 \\
\lambda_2\pi_2 + \lambda_1\pi_5 - (\lambda_1+\lambda_2+\mu)\pi_6 + \mu\pi_7 = 0 \\
\lambda_2\pi_3 + \lambda_1\pi_6 - (\lambda_1+\lambda_2+\mu)\pi_7 + \mu\pi_8 = 0 \\
\lambda_2\pi_4 + \lambda_1\pi_7 - (\lambda_2+\mu)\pi_8 = 0 \\
\lambda_2\pi_5 - (\lambda_1+\lambda_2)\pi_9 + 2\mu\pi_{10} = 0 \\
\lambda_2\pi_6 + \lambda_1\pi_9 - (\lambda_1+\lambda_2+2\mu)\pi_{10} + 2\mu\pi_{11} = 0 \\
\lambda_2\pi_7 + \lambda_2\pi_8 + \lambda_1\pi_{10} - (\lambda_1+\lambda_2+\mu)\pi_{11} + 2\mu\pi_{12} = 0 \\
\lambda_1\pi_{11} - (\lambda_2+2\mu)\pi_{12} = 0 \\
\lambda_2\pi_9 - \lambda_1\pi_{13} + 3\mu\pi_{14} = 0 \\
\lambda_2\pi_{10} + \lambda_1\pi_{13} - (\lambda_1+3\mu)\pi_{14} = 0 \\
\lambda_2\pi_{11} + \lambda_1\pi_{14} - (\lambda_1+\mu)\pi_{15} + 3\mu\pi_{16} = 0 \\
\lambda_2\pi_{12} + \lambda_1\pi_{15} - 3\mu\pi_{16} = 0 \\
\sum_{i=1}^{16} \pi_i = 1
\end{cases} \tag{10}$$

Thus, we obtain the stationary probabilities:

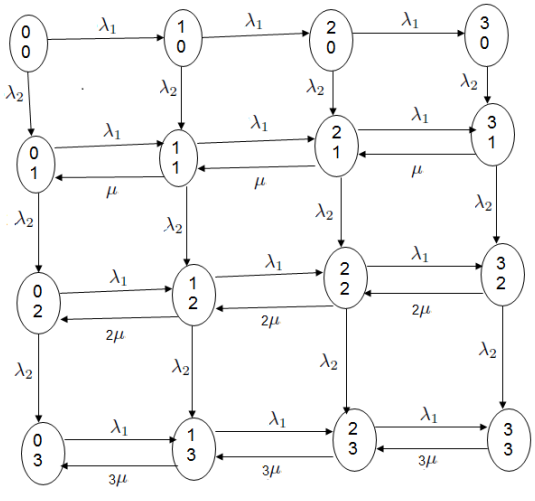
$$\begin{cases} \pi_1 = 0 \\ \pi_2 = 0 \\ \pi_3 = 0 \\ \pi_4 = 0 \\ \pi_5 = 0 \\ \pi_6 = 0 \\ \pi_7 = 0 \\ \pi_8 = 0 \\ \pi_9 = 0 \\ \pi_{10} = 0 \\ \pi_{11} = 0 \\ \pi_{12} = 0 \\ \pi_{13} = \frac{27\mu^3}{27\mu^3+9\mu^2\lambda_1+3\mu\lambda_1^2+\lambda_1^3} \\ \pi_{14} = \frac{9\lambda_1\mu^2}{27\mu^3+9\mu^2\lambda_1+3\mu\lambda_1^2+\lambda_1^3} \\ \pi_{15} = \frac{3\lambda_1^2\mu}{27\mu^3+9\mu^2\lambda_1+3\mu\lambda_1^2+\lambda_1^3} \\ \pi_{16} = \frac{\lambda_1^3}{27\mu^3+9\mu^2\lambda_1+3\mu\lambda_1^2+\lambda_1^3} \end{cases} \tag{11}$$

We assume that a general form of the stationary probabilities is defined as following: $\pi = (\pi_1, \pi_2, \pi_3, \pi_{(n+1)^2})$

For $n = m = 1$, we have

$$\begin{cases} \pi_1 = 0, \\ \pi_2 = 0, \\ \pi_3 = \frac{\mu}{\lambda_1+\mu}, \\ \pi_4 = \frac{\lambda_1}{\lambda_1+\mu}. \end{cases} \tag{12}$$

Figure 8 Markov chain of the Petri nets model for $n = m = 3$



For $n = m \geq 2$, we have

$$\begin{cases} \pi_i = 0, & \text{if } 1 \leq i \leq (n+1)^2 - (n+1), \\ \pi_i = \frac{(n\mu)^{(n+1)^2-i}\lambda_1^{n-[(n+1)^2-i]}}{(n\mu)^n + \lambda_1^n + \sum_{j=1}^{n-1}(n\mu)^j(\lambda_1^{n-j})}, & \text{if } (n+1)^2 - n \leq i \leq (n+1)^2. \end{cases} \tag{13}$$

Using the stationary probabilities, we calculate the average numbers of clients requests in the system L_s is:

$$L_s = \sum_{i=1}^{(2n)^2} i\pi_i$$

$$\begin{cases} L_s = 3\pi_3 + 4\pi_4, & \text{if } n = m = 1, \\ L_s = \sum_{i=(n+1)^2-n}^{(n+1)^2} i\pi_i, & \text{if } n = m \geq 2. \end{cases} \tag{14}$$

$$\begin{cases} L_s = \frac{3\mu+4\lambda_1}{\lambda_1+\mu}, & \text{if } n = m = 1, \\ L_s = \sum_{i=(n+1)^2-n}^{(n+1)^2} \frac{i(n\mu)^{(n+1)^2-i}\lambda_1^{n-(n+1)^2+i}}{(n\mu)^n + \lambda_1^n + \sum_{j=1}^{n-1}(n\mu)^j\lambda_1^{n-j}}, & \text{if } n = m \geq 2. \end{cases} \tag{15}$$

According to the formula of little:

$$W_s = \frac{L_s}{\lambda_1}$$

From where average time of stay W_s is:

$$\begin{cases} W_s = \frac{3\mu+4\lambda_1}{\lambda_1(\lambda_1+\mu)}, & \text{if } n = m = 1 \\ W_s = \sum_{i=(n+1)^2-n}^{(n+1)^2} \frac{i(n\mu)^{(n+1)^2-i}\lambda_1^{n-(n+1)^2+i}}{\lambda_1[(n\mu)^n + \lambda_1^n + \sum_{j=1}^{n-1}(n\mu)^j\lambda_1^{n-j}]}, & \text{if } n = m \geq 2 \end{cases} \tag{16}$$

From the relations, we remark that arrival rate of web services does not influence average time of stay in the system, because each time client requests finish and quit service, the web services return in their station, i.e., the web services rejoin their queues, to be used by other client requests.

5.3 Numerical application

In order to calculate response time, we assume that the inputs criteria are: $\lambda_2 = 0.01$, $\mu = 1$, the different values of λ_1 are: $\lambda_1 = 0.00001, 0.0001, 0.001, 0.01, 0.1$ and the different values of n are $n = 1, 10, 100, 10{,}000, 100{,}000, 1{,}000{,}000, 10{,}000{,}000$.

The obtained results are presented in Table 2.

Table 2 Average response time

	$\lambda_1 = 0.00001$	$\lambda_1 = 0.0001$	$\lambda_1 = 0.001$	$\lambda_1 = 0.01$	$\lambda_1 = 0.1$
$n = 1$	$3 * 10^5$	$3.0001 * 10^4$	$3.001 * 10^3$	$3.0099 * 10^2$	30.909
$n = 10$	$99.9001 * 10^5$	$99.9010 * 10^4$	$99.9101 * 10^3$	$100.0009 * 10^2$	1,009.183
$n = 100$	$10^4 * 10^5$	10^8	10^7	10^6	$1.0010 * 10^5$
$n = 120$	$1.44 * 10^9$	$1.44 * 10^8$	$1.44 * 10^7$	$1.4401 * 10^6$	$1.4412 * 10^5$
$n = 140$	$1.96 * 10^9$	$1.96 * 10^8$	$1.96 * 10^7$	$1.9601 * 10^6$	$1.9614 * 10^5$
$n = 141$	$1.9881 * 10^9$	$1.9881 * 10^8$	$1.9881 * 10^7$	$1.9882 * 10^6$	$1.9895 * 10^5$
$n = 142$	∞	∞	∞	∞	∞

Table 2 presents average response time according to n, for the different values of the arrival rate of client request λ_1.

From Table 2, we remark that each time λ_1 increases, average response time increases. Because the number of clients requests who arrive increases. Each time n increases until $n = 141$, average response time increases, because we have many web services, and the clients will make all possibility for discovery and composition of web services, and will take more time. We costate that for $n \geq 142$, average response time is ∞, the system becomes saturated.

Table 3 presents average number of client requests in the system according to n, for the different values of the arrival rate of client request λ_1.

Table 3 Average number of client requests in the system

	$\lambda_1 = 0.00001$	$\lambda_1 = 0.0001$	$\lambda_1 = 0.001$	$\lambda_1 = 0.01$	$\lambda_1 = 0.1$
$n = 1$	3	3.0001	3.001	3.0099	3.0909
$n = 10$	99.9001	99.9010	99.9101	100.0009	100.9183
$n = 100$	10^4	10^4	10^4	10^4	$1.0010 * 10^4$
$n = 120$	$1.44 * 10^4$	$1.44 * 10^4$	$1.44 * 10^4$	$1.4401 * 10^4$	$1.4412 * 10^4$
$n = 140$	$1.96 * 10^4$	$1.96 * 10^4$	$1.96 * 10^4$	$1.9601 * 10^4$	$1.9614 * 10^4$
$n = 141$	$1.9881 * 10^4$	$1.9881 * 10^4$	$1.9881 * 10^4$	$1.9882 * 10^4$	$1.9895 * 10^4$
$n = 142$	∞	∞	∞	∞	∞

From Table 3, we remark that each time λ_1 increases and the average number of clients requests in the system increases, because the number of clients requests who arrive increases. For $n \geq 142$, the system becomes saturated.

6 Position

In the recent few years, the number of web services discovery and composition techniques proposed in the literature, have remarkably increased and web services performance analysis has become an important research axis. Because of most complex tasks on web services discovery and composition, web service that still fail to raise a software agent's satisfaction, new providers should be able to offer suitable web services.

Table 4 Our proposition in the comparative study of works on the evaluation of performance of the web services

		Our proposition
Model	Queue networks	
	Markov chain	x
	Linear programming	
	Heuristic approach	
	Optimisation approach	
	Stochastic automata networks	
	Petri nets	x
Solution	Analytical solution	x
	Simulation	
	Prototype	
Qos attributs	Capacity	
	Availabiliy	
	Reliability	
	Response time	x
	Cost	
Other performances	Consistency	
	Statistical performance estimates	
	Number of clients and web services	x
	Composition time	
	Turnaround time	
	Waiting time	
	Service time	
Web services	Simple	
	Composite	x

In this research work, first, we have adopted to an analytical model, Petri nets for the system of web services, taking into account clients and web services arrivals. Secondly, we have resolved it analytically and we calculate average response time, in terms of the arrival rate of the clients requests, and we obtain the number limit of web services to have saturation of the system.

We illustrate our proposition in the comparative study of works on the evaluation of performance of the web services in Table 4. Our model takes into account two types, web services and requests, and we have made a relation between these two types of arrival, while other works have taken into account web services composition methods and their architecture, we modelise the system with a Petri nets model, and to have a performance of this model we used an analytical method.

7 Conclusions

The evaluation of quantitative measures characterising the clients requests is widely recognised as highly important to faithfully reflect the impact of failures and performance degradation. Few researches have been devoted to the analysis of the system of web services taking into account clients and web services requests. These

studies are based on measurements and analytical modelling of the clients requests and web services. However, there is still a need for measurements and analytical modelling request satisfaction in composite web services, to make early predictions supporting the design of such systems.

Our contribution consists in modelling such systems with stochastic Petri nets. This approach allowed us to have an analytical model of the web services system that takes into account two different types of arrivals: clients and web services. The performances of this model are calculated using an analytical method. We got the limit number of clients requests and web services in the system to be saturated, which is 142. However, this work may take into consideration other aspects: so we can take sure that clients requests and web services are different, and that such a client wants such a web service, and we can specified that at the end of the service, the clients can be satisfied and leave the system, or not satisfied and attempt another attempt and modelled the system with the coloured Petri nets and the resolution of this model analytically. Furthermore, we propose to do a study, taking into account the manner which the web services are presented in the system for example parallel, series.

References

Abdulqader, O.H. and Nasruddin, H. (2018) 'Pareto optimal solution for multiobjective stochastic linear programming problems with partial uncertainty', *International Journal of Mathematics in Operational Research*, Vol. 12, No. 2, pp.139–166.

Ansorena, I.L. (2019) 'Work planning optimisation in ports: a simplex application', *IEEE International Journal of Mathematics in Operational Research*, Vol. 14, No. 1, pp.146–155.

Bjork, K.M. and Mezei, J. (2016) 'A heuristical solution method to separable non linear programming problems', *International Journal of Mathematics in Operational Research*, Vol. 9, No. 3, pp.230–242.

Boustil, A. et al. (2016) 'A semantic selection strategy for composite web services based on conforming objects', *International Conference on Information Technology for Organizations Development (IT4OD)*, Fez, Morocco.

Boutrous Saab, C. et al. (2006) 'Performance evaluation for mobile access to composite web services', *Proceeding of International Conference on Internet and web Applications and Services (ICIW'06)*, Gosier, France, pp.154–159.

Chandrasekaran, S. et al. (2003) 'Performance analysis and simulation of composite web services', *Taylor and Francis Group Journal*, Vol. 13, No. 2, pp.120–132.

Chattopadhyay, S. and Banerjee, A. (2017) 'QoS constrained large scale web service composition using abstraction refinement', *IEEE Transactions on Services Computing*, No.99, p.1, DOI: https://doi.org/10.1109/TSC.2017.2707548.

Chemaa, S. et al. (2012) 'A high-level Petri net based approach for modeling and composition of web services', *Proceedings of the International Conference on Computational Science (ICCS)*, Omaha, Nebraska, pp.469–478.

Comet, J.P. (2014) *Les réseaux de Petri pour la simulation de systémes biologiques*, Laboratoire I3S, UMR 6070 CNRS/UNSA Université de Nice-Sophia-Antipolis Ecole Polytech.

Csizmar, D.A. (2001) 'An optimal service ordering for a world wide web server', *ACM SIGMETRICS Performance Evaluation Journal*, Vol. 29, No. 2, pp.8–13.

Deng, Z. et al. (2016) 'A reliability calculation method for web service composition using fuzzy reasoning colored Petri nets and its application on supercomputing cloud platform', *Future Internet Journal*, Vol. 8, No. 4, p.47.

Ding, Z. et al. (2016) 'Performance evaluation of transactional composite web services', *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 46, No. 8, pp.1061–1074.

Fensel, D. et al. (2002) 'Semantic web enabled web services', *Sigmod Record ACM Journal*, Vol. 31, No. 4, pp.1–2.

Gardarin, G. (2002) *XML des bases de donnees aux Services Web*, in French, Dunod, Paris.

Garima, G. et al. (2017) 'Performance evaluation of composed web services', *International Journal of Computer Applications*, Vol. 171, No. 4, pp.975–8887.

Haddad, S. et al. (2013) 'Bounding models families for performance evaluation in composite web services', *Computational Science Journal*, Vol. 4, No. 4, pp.232–241.

Haddadi, S. et al. (2018) 'A two-phase heuristic for set covering', *International Journal of Mathematics in Operational Research*, Vol. 13, No. 1, pp.61–78.

Hoecke, S.V. et al. (2005) 'Modelling the performance of the web service platform using layered queueing networks', *Proceedings of the 27th International Conference on Software Engineering Research and Practice (SERP'05)*, St. Louis, USA.

Jun, L. and Jian, L. (2019) 'Research on the influence of sampling methods for the accuracy of web services QoS prediction', *IEEE Access Journal*, Vol. 7, pp.39990–39999, DOI: https://doi.org/10.1109/ACCESS.2019.2906769.

Klein, A. et al. (2010) 'Efficient QoS-aware service composition with a probabilistic service selection policy', *Service-Oriented Computing Journal*, Vol. 6470, pp.182–196, DOI: https://doi.org/10.1007/978-3-642-17358-5_13.

Kumar, M. (2015) 'Various factors affecting performance of web services', *International Journal of Sensor and Its Applications for Control Systems*, Vol. 3, No. 2, pp.11–20.

Matteo, G. et al. (2019) 'An active learning approach to build adaptive cost models for web services', *Data and Knowledge Engineering Journal*, Vol. 119, pp.89–104, DOI: https://doi.org/10.1016/j.datak.2019.01.001 .

Mecheri, K. et al. (2019) 'Context-based interoperability of semantic web services', *International Journal of Metadata, Semantics and Ontologies*, Vol. 13, No. 3, pp.209–226.

Mohan Reddy, C.R. et al. (2013) 'QOS of web service: survey on performance and scalability', *Computer Science and Information Technology (CS and IT)*, Vol. 3, pp.65–73, DOI: https://doi.org/10.5121/csit.2013.3907.

Mokdad, L. et al. (2015) 'Bounding models families for performance evaluation in composite web services', *Proceedings of the 1st International Conference on Industrial Networks and Intelligent Systems (INIS'15)*, Tokyo, Japan.

Nacer, H. and Aissani, D. (2014) 'Semantic web services: standards, applications, challenges and solutions', *Network and Computer Applications Journal*, Vol. 44, pp.134–151. DOI: https://doi.org/10.1016/j.jnca.2014.04.015.

Nacer, H. et al. (2009) 'Semantic annotations for web services discovery and composition', *Computer Standards and Interfaces Journal*, Vol. 31, No. 6, pp.1108–1117.

Nacer, H. et al. (2017) 'A distributed authentication model for composite web services', *Compter and Security Journal*, Vol. 70, pp.144–178, DOI: https://doi.org/10.1016/j.cose.2017.05.008.

Naseri, A. and Navimipour, N.J. (2019) 'A new agent-based method for QoS-aware cloud service composition using particle swarm optimization algorithm', *Journal of Ambient Intelligence and Humanized Computing*, Vol. 10, No. 5, pp.1851–1864.

Niu, S. et al. (2019) 'Towards the optimality of QoS-aware web service composition with uncertainty', *International Journal of Web and Grid Services*, Vol. 15, No. 1, pp.1–28.

Pinto, A.A. et al. (2017) 'A performance evaluation of an automatic web services composition system', *International Conference on High Performance Computing and Simulation (HPCS)*, Genoa, Italy, pp.839–845.

Reddy, C.R.M. et al. (2011) 'Early performance prediction of web services', *International Journal on Web Service Computing*, Vol. 2, No. 3, pp.31–41.

Reza, H. et al. (2018) 'An a-cut approach for fuzzy product and its use in computing solutions of fully fuzzy linear systems', *International Journal of Mathematics in Operational Research*, Vol. 12, No. 2, pp167–189.

Rosenber, F. et al. (2010) 'Metaheuristic optimization of large-scale QoS-aware service compositions', *Proceedings of the IEEE International Conference on Services Computing*, Miami, Florida, pp.97–104.

Sam, M.L. et al. (2018) 'Comparison between linear programming and integer linear programming: a review', *International Journal of Mathematics in Operational Research*, Vol. 13, No. 1, pp.91–106.

Sato, N. and Trivedi, K.S. (2007) 'Stochastic modeling of composite web services for closed-form analysis of their performance and reliability bottlenecks', *Proceedings of the ICSOC'07, The 5th International Conference on Service-Oriented Computing*, Vienna, Austria, pp.107–118.

Singh, R.P. and Pattanaik, K.K. (2013) 'An approach to composite QoS parameter based web service selection', *Proceedings of the 4th International Conference on Ambient Systems, Networks and Technologies (ANT 2013)*, Halifax, Nova Scotia, Canada, pp.470–477.

Wang, P. et al. (2006) 'A fuzzy model for selection of QoS-aware web services', *Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE'06)*, Shanghai, China, pp.585–593.

Weitao, H. and Wei, Z. (2018) 'Reliability prediction and QoS selection for web service composition', *Journal of Computers*, Vol. 29, No. 5, pp.177–189.

Yugen, D. and Yonggang, L. (2010) 'Virtual web services and its application in e-commerce', *Proceeding of the 2nd IEEE International Conference on Information Science and Engineering (ICISE)*, TBD, Hangzhou, China, pp.1–5.

Zheng, Z. et al. (2017) 'Semi-Markov models of composite web services for their performance, reliability and bottlenecks', *IEEE Transactions Services Computing*, Vol. 10, No. 3, pp.448–460.

Zhong, D. and Qi, Z. (2006) 'A Petri net based approach for reliability prediction of web services', *Proceedings of OTM Workshop*, Montpellier, France, pp.116–125.

Notes

1 XML is a markup language to describe different data types.