Sudan University of Science and Technology

The Collage of Computer Science and Information Technology

Validity of QoS Requirements for Application Using Probabilistic Model Checking (Case Study: Web Applications)

التحقق من جودة خدمة متطلبات التطبيق باستخدام نموذج الاختبار الاحتمالي (دراسة حالة: تطبيقات الانترنت)

Prepared for award master degree by:-

HANAN AHMED ALI AHMED

Supervised by:-

Dr. Nadir Kamal Salih

March 2019

Acknowledgement

I would like to thank my advisor to support me and quest to find out what is the best, and thank all staff in the faculty of computer science and technology and my manager at work to support me to continue.

I would like to thank my family: my parents and sister for supporting me spiritually throughout writing this thesis and my life in general.

I would like to thank my friends who encourage me to overcome the obstacles.

Abstract:

In the field of computer applications when be there need for design application to present certain service for specific customer, this application must design in care of how satisfy customer needs, therefore any application must evaluate if satisfy all customer needs or has violation in some needs. Customer needs is customer requirements, and customer satisfaction level sometimes called Quality of Services (QoS), so the customer requirements are main factor use to evaluate QoS. The customer requirements differ according to the type of service.

The study was initiated to solve the problem of validity of QoS requirements for application, it is to evaluate if the application satisfy user requirements or not, the user requirements is considered as QoS requirements because it represented according to agreements between customer and service provider which called service level agreement (SLA), the violation in one of QoS requirements means there defect in application, and must redesign of application.

To validate QoS requirements the study use probabilistic model checking, this technique depends on probabilistic techniques and model checker software. The probabilistic technique was used to represent system states and their requirements which represented as probability values, the system states represented using Markov chains, these states is represented according to system navigation, mean when system transfer from certain state to another and probability of navigation. The system quality requirements represented using Discrete Time Markov Chain (DTMC) and Continuous Time Markov Chain (CTMC) which specify according to time state, the requirements represented in DTMC transform to Probabilistic Computation Tree Logic (PCTL) formula, and another one transform to Continuous Stochastic Logic (CSL) formula. Then use software PRISM as model checker, in software represent system states and probabilities in modules using model language, and then run module for each one to gain results which specify if the system achieve user satisfaction or not.

In this study used the web application used as case study and after follow proposed solution, the outputs of experiments clarify that some of requirements were violated and then the system need to redesign to achieve QoS requirements.

A future studies must concern about how the validity of QoS starts at the earlier stages, and must do at parallel with the design of application to avoid increase of cost for redevelop the applications.

ملخص البحث:

في مجال تطبيقات الحاسوب,عندما تكون هناك حوجة الى تصميم تطبيق بغرض تقديم خدمة محددة الى عملاء محددين. مثل هذه التطبيقات عندما تصمم من قبل المختصين يجب ان يكون في الاعتبار تلبية جميع احتياجات العميل من الخدمة التي من اجلها صمم التطبيق. للتأكد من ان التطبيق يلبي جميع احتياجات العميل يجب ان تكون هناك طريقة لتقييم ما اذا كان التطبيق يقوم بالمهمة التي صمم من اجلها بالقدر الذي المطلوب, ام ان هناك قصور في بعض جوانب التطبيق لا يلبي فيها جميع احتياجات العميل. احيتاجات العميل عبارة عن متطلبات العميل من الخدمة او التطبيق , بينما مستوى تلبيه هذه المتطلبات عبارة عن مقياس لمدى جودة الخدمة او التطبيق. بالتالي متطلبات العميل هي العامل الساسي الذي يستخدم لتقييم جودة الخدمة.

هذه الدراسة انشئت للبحث عن حل لمشكلة التحقق من متطلبات الجودة للتطبيق, وهنا التحقق من ماذا كان التطبيق يلبي جميع متطلبات المستخدم ام لا. متطلبات المستخدم عبارة تعتبر متطلبات الجودة للتطبيق لانها تمثل وفقا للاتفاق بين طالب الخدمة (العميل) ومقدم الخدمة , اذا كان هناك قصور في تلبية احد متطلبات العميل من الخدمة هذا يعني ان هناك مشكلة في جودة النظام مما يلزم البحث سبب المشكلة وحلها.

للتحقق من متطلبات الكفاءة للنظام تم استخدام تقنية التحقق من النموذج الاحتمالي, وهذه التقنية التقنية تعتمد على الاحتمالية وبرنامج التحقق من النموذج model checker . تستخدم تقنية الاحتمالية لتمثيل النظام في حالاته المختلفة بحيث اي حالة يتم الانتقال اليها من حالة قبلها باحتماليه معينة وهذه ماتسمى بمتطلبات جودة النظام . حالات النظام يتم تمثيلها باستخدام سلاسل ماركوف المعروفة بالقدرة على تمثيل سلوك النظام باحتمالات التنقل بين حالاته. متطلبات النظام يتم تقسيمها حسب ارتباطها بالزمن بحيث بعض حالات النظام ترتبط بفترة زمنية محددة وهذه تمثل في شكل CTMCلتمثيل الفترة الزمنية التي تستغرها حالة معينة من حالات النظام قبل الانتقال الى غيرها , اما بعض الحالات فتظهر دون الارتباط بالزمن وهذه تمثل في صورة DTMC . متطلبات النظام التي تمثل في صورة DTMC يعبر عنها في صورة PCTL والتي تمثل في صورة CTMC يعبر عنها في صورة CSL. بعد ذلك يستخدم برنامج التحقق من النموذج للتحقق من متطلبات جودة النظام وهو مايسمى PRISM. في هذا البرنامج يتم تمثيل حالات النظام واحتمالاتها باستخدام لغة النمذجة ثم تنفيذ النوذج للحصول على نتائج التحقق وتحديد ما اذا كان النظام يحقق متطبات الكفاءة ام ان هناك قصور(انتهاك) لمتطبات الكفاءة.

في هذه الدراسة لتطبيق الحل المقترح تم اختيار تطبيق انترنت كحالة دراسة.

النتائج المتحصل عليها وضحت ان بعض متطلبات الجودة تم تحقيقها بينما البعض لم يتم تحقيقها ممايلزم اعادة تطوير النظام مرة اخرى لتحيق متطلبات الجودة.

من هذه الدراسة نخلص الى اهمية تحقيق متطلبات الجودة للانظمة وضرورة العمل في مراحل مبكرة من التصميم لتجنب التكلفة الاضافية في اعادة التطوير.

Preliminary

Chapters Contents

List of tables

List of figures

List of Abbreviations:

Index	Abbreviation	Meaning
1.	**QoS**	Quality of Service
2.	**PCTL**	Probabilistic Computation Tree Logic
3.	**CSL**	Continuous Stochastic Logic
4.	**DTMC**	Discrete Time Markov Chain
5.	**CTMC**	Continuous Time Markov Chain

List of Appendices

CHAPTER I

INTRODUCTION

1-1-Introduction:

Quality of service (QoS) has been receiving wide attention in the recent years in many research communities including networking, multimedia systems, real-time systems and distributed systems. These systems have a requirement that applications contending for system resources must satisfy timing, reliability and security constraints as well as application-specific quality requirements, these requirements are specifying according to the QoS parameters [1].

The QoS parameters will specify considering systems communities, The QoS parameters values standard depend on the agreements between service provider and the customer , the agreements called service-level agreements (SLA), it represents user satisfaction threshold and used to decide if the service met user requirements. From all mentioned, QoS is the concept of specifying how good the offered services are. It provides means to evaluate services. To evaluate services the QoS parameters must measure, the measurement of QoS parameter may achieve using deferent techniques, the QoS parameter will specified according to communities need to evaluate systems belong. According to purpose of QoS measurement, the QoS parameters specified.

The process of QoS evaluation was becomes important activity, because is contributing on another activities and help to make decision about services which considered in the evaluation. These activities or can say the purposes of QoS evaluation are: measure user satisfaction level from service, improve the service and discover the violation on it, to select between services has same functionality, and to evaluate if the service meet the requirements intended for.

The purpose which considered in this research, is the evaluation service to decide if it is meet user requirements, these requirements represents the QoS, for assurance a service satisfies the QoS requirement must use techniques for validity. The validity is QoS evaluation activity shows that the solution fulfills the requirements and service level agreement are met. The implementation of component is acquired, that the implementation of is match the specifications and accuracy is ensured by a transition to the activity [2].

1-2-The Problem Statement:

The main problem was solved through this study is the validity of QoS requirements for application. Any application was design to present service at specific field; applications always intended after specify services which need to achieve, so there must be clear objectives gained from application, the objectives expressed through list of requirements. To design application take into account the requirements, the requirements are general concepts involve all important point of application, like environment which application run in it, and type of service which intended for it standard, these requirements represent QoS

parameters which may need to evaluate application from QoS point of view. The QoS evaluation for application may need for multi purposes.

The QoS evaluation contribute to solve multi problems, one of these problem is the validation of application if meet the requirements intended according to it, also if there more than one application has same functionality, the measurement of QoS help to choose the optimal one, another problem to solve depend on QoS evaluation, if the user need to certain services not available by one application, like this services available cross composition more than one application compose together to provide required features, the optimal composition is choose according results of QoS measurements.

In this study was focused on validity of QoS requirements for application to test if it was meet the QoS requirements using web application as case study.

1-3-Significant of Research:

the process of validity of QoS requirement is an important process to verify if the application achieves the services which intended for it according to QoS requirements and gain more accurate result from validity which help in optimize the service level and support the correctness process for application, because it help in discover the violation in QoS if exist and specify this violation.

1-4-Research Objective:

QoS is wide research area because is popular concept, and used in many field to express about service quality level, many researches was published about QoS, each one has different view for QoS and establish the study for certain purposes. Below will mention our research objectives:

1. To build model for check QoS for application.
2. To verify if the application achieve required QoS.
3. To Discover QoS requirements violation.

1-5-Scope:

Will use web application as a case study for achieving research objectives, furthermore the scope of research is requirements of web application, and the requirement will specify according to application itself and environments which run because it is effect on the running application.

1-6-Research Outline:

Chapter one: Research Background

Chapter two: Literature Review

Chapter three: Methodology and research framework

Chapter four: Implementation

Chapter five: Conclusion

CHAPTER II

Literature Review

2-1-Introduction:

Quality of service (**QoS**) is the description or measurement of the overall performance of a service, such as telephony or computer network or Cloud computing service, particularly the performance seen by the users of the network. To quantitatively measure quality of service, several related aspects of the network service are often considered, such as error rates, bit rate, throughput, transmission delay, availability, jitter, etc.

According to the International Telecommunication Union (ITU-T) standard E.800, QoS is defined as "the collective effort of service performances, which determine the degree of satisfaction of a user of this service"[4]. QoS is also defined as "a set of service requirements to be met by the network while transporting a flow". Services can have qualitative and quantitative QoS parameters [5]. Qualitative parameters are security mechanisms, manageability, *etc.* Quantitative parameters are bandwidth, jitter, delay, *etc.* These parameters together determine the Quality of Service (QoS).
The QoS is a wide research area, and more researches was published about it, According to purpose of published researches can classify these studies to three topics: first purpose is services evaluation, second purpose is to help customers to select between applications which provide the same services, and the customer select the services has an advantage over other services that do not provide this features [3], third purpose is services composition, this purpose for composite between group of services when there is no one service fulfill customer requirements.

Here will present number of these published studies and classifications, brief point and analyze these studies; contributions and QoS aspects which concerned by researchers. Also talk about techniques used to solve problem of study.

2-2 Techniques and Tools:

In this section will talk in details about techniques and tools used in the proposed solution

1-Probabilistic techniques:

Definitions: You know the likelihood that something will happen, but you don't know when it will happen. Also can define as:

1. Are techniques can be used to modeling, specifying, and proving timing properties of real-time systems [41].
2. Are formal verification techniques for modeling and analysis of systems that exhibit stochastic behavior [42].

3. Probabilistic is an effective technique to prove the presence combinatorial objects having some specific properties [43].

The probabilistic techniques has many types each one used at certain constrain and condition here will demonstrate temporal logic which has many branches.

2-Temporal Logic:

Temporal logics allows you to specify the behavior of systems in terms of logical formulas, including time constraints, events, and relationships between the two [41]. The temporal logics were mathematically founded, so these techniques facilitate modeling, specifying, and proving timing properties of real-time systems, also have different capabilities for the specification, validation, and verification [44].

The temporal logics have two extension areas real-time and probability, here will concern about probability. The probability mean reason about behavior of a program under probabilistic assumptions about the input, reason about uncertain information in an expert system or deal with dependability requirements, the dependability requirements indicate that the probability of undesirable but unsustainable behavior of the system must be less than a certain limit.

Temporal logics have multi types; in our model will use combination branching time temporal logic called computation tree logic (CTL) and probabilistic temporal logic, Probabilistic Computation Tree Logic (PCTL), and another one also is computation tree logic is continuous stochastic logic (CSL).

3-Probabilistic Computation Tree Logic:

Probabilistic Computation Tree Logic is a probabilistic branching time temporal logic which allows for probabilistic quantification of described properties. In PCTL, one can also associate a time bound with a path operator, so is a useful logic for stating soft deadline properties [45]. Also it used to express dependable properties such as reliability and availability [46].

PCTL is used to represent properties of application at discrete state; this means the application state change at deferent point of time [47], one time unit corresponds to one transaction along an execution path. Discrete time Markov chain (DTMC) interpreted PCTL formulas. The PCTL has state formulas and path formulas, state formulas represent states properties while path formula represents paths properties (sequence of states). At later section will present Markov chains concepts. **Discrete Time Markov chain (DTMC)** used to model systems whose behavior can be described at each time point by a separate probabilistic option across several possible outcomes. So it can translate a system to state-

transition where each transition is added with a potential probability value [48].Discrete time Markov chain consists of discrete states, representing the configurations of the system, and has transitions governed by (discrete) probability distributions over the target states.

4-Continuous Stochastic Logic (CSL):

CSL is a branching-time temporal logic CTL, is widely used in analyzing practical system performance and reliability [49]. It includes means to evident transient and steady-state performance measures [50]. CSL is a powerful logic for expressing quantitative time-bounded constrained reachability properties.

CSL as PTCL has state formula and path formula. To represent CSL the continuous time Markov chain (CTMC) used. To illustrate the concept of CTMC, suppose the system enters state i at time $t \geq 0$: It stays in state i for a random amount of time called the sojourn time and then jumps to a new state $j \neq i$ with probability p_{ij} . The sojourn time and the new state depend only on i and not on the history of the system prior to time t. Furthermore, given that the current state is i; the sojourn time and the new state are independent of each other [51].

5-Probability Assignment Techniques:

The probability of an event is a numerical measure of the chance that this event will occur.

The probability value of event must calculate according to below condition:

1. A given statistical experiment has at least two experimental outcomes.
   Depending on the problem being analyzed, either some of the outcomes, or even all of them could be events of statistic interest

2. The probability of any event e_i if denote it $P(e_i)$ must lie in range between 0 and 1:
   $$0 \leq P(e_i) \leq 1$$
3. The sum of the probabilities of all n experimental outcomes equals 1 :
   $$P(e_1)+ P(e_2)+ P(e_3)+\ldots\ldots+ P(e_n)=1.$$

To assign probability values there three approaches of assigning, those approaches depend on the nature of the information that is available, below will mention details about each approach:

1-The Classical method for assigning probability:

If probabilities of the experimental outcomes satisfy the following assumptions:

a) The probabilities of all of the outcomes are known in advance, and

b) The outcomes are equiprobable (all the outcomes are equally likely). Then the probability of each of then outcomes is1/n.... then we can deal with so called classical method of assigning probabilities.

2-Relative Frequency method of assigning probability:

When the assumption that the outcomes of a statistical experiment are known in adv ance

and are equally likely is not satisfied, the estimation of probability for events of inte rest can be done by using past statistics.

3-subjective method:

This method is used when the assumptions used in the classical method are not appl icable

and the past statistics that can be used for the relative frequency method are unavailable.
In such a situation, the basis for assigning probability to experimental outcomes is previous business experience, belief, and even feeling.
Since this method relies on individual judgmental, it is highly subjective. Nonetheless, the method is quite common in the absence of any information. Later, when the information becomes available, the assigned probability can be revised.
In this study was using the third approach.

6-Markov chain:

Definition:

The term "Markov chain" refers to the sequence of random variables such a process moves through, with the Markov property defining serial dependence only between adjacent periods (as in a "chain").It can therefore be used to describe systems that follow a series of related events, where only what happens next depends on the current state of the system. Markov chain is stochastic model used to describing sequence of possible events in which the probability of each event influence by previous event, so the Markov process used to predict future events depend on current state (event) [52].

Markov chains can characterized as follow: each system have set of states S= $\{s_1, s_2, s_3, \ldots\ldots\ldots s_n\}$ the set is state space, the transition from one state to another is called step, the probability of what is the next step called transition probability. The Markov chain is a type of Markov process in either discrete or continuous time.

The model checker will use for verification of QoS requirements is software called PRISM.

7- Probabilistic Model Checking:

Probabilistic Model Checking is an automatic model-based verification approach used to analyze probabilistic system and explores all system executions, the analysis will use in correctness, performance and reliability of system [53].

To analyze probabilistic system using model checking, the inputs are a probabilistic model and a probabilistic property described in a specification language, and the output is whether or not the model satisfies the property, the probabilistic property specification describes using PCTL and CSL [54].

The tools used for checking are software developed for the purpose of verification; this research will use one of this software named PRISM.

It is an open source probabilistic model checker developed initially at the University of Birmingham and now at the University of Oxford. It is supports three probabilistic models: discrete-time Markov chains (DTMC), continuous-time Markov chains (CTMC) and Markov decision processes.[55], these models are described in the PRISM modeling language, a relatively simple, state-based language and properties are specified in a logic which incorporates LTL, PCTL, CSL.

2-3-Advantages of used technique over another:

The techniques will used is automotive verification, this mean the verification performed without human intervention, the implementation of model checker is clear the strangest of this technique compared with simulation-based technique. When it was used as a debugging tool, the bugs detected in arbiter could not detect with simulation tool. The automotive verification now use in many communities like industrial practices, networking, real-life systems and small device.

2-4-QoS Requirements for applications:

Quality of service is a very popular and overloaded term that is very often looked at from different perspectives by the networking and application-development communities [7]. Any application has its own parameters which used to evaluate performance of it , these parameters must be at certain level to meet the user satisfaction that called OoS requirements, these QoS requirements vary from one application to another so the parameters vary from one application to another.

To define the Quality of Service a customers of a services need to establish service level agreements (SLA)[8]. At below sections describe QoS requirements for each type of application. Many network applications work with Best-Effort services, while others have

strong QoS requirements and only work with guaranteed QoS, or at least benefit significantly if QoS guarantees are possible. Here will give an overview of application requirements for audio, video and data applications [9].

1-Audio Applications:

Audio applications QoS requirement parameter which called bandwidth, delay and packet loss determined according to audio transmission type, like telephony and high fidelity music.in addition to above factor. The bandwidth also affected by the encoded audio data, protocol overhead by IP, User Data Datagram (UDP), and Real-Time Transport Protocol (RTP) headers. The delay specified according to sensitivity of transmission to delay such as telephony strong delay requirement exists.

2-Video Applications:

Similar to audio applications, but there several deferent between audio and video applications, video require much higher bandwidth depend on quality level required by a user or supported by equipment (PC and mobile hand held).

3-Data Applications:

Nonvideo and nonaudio application classified as data application, there multi type of data applications each of it different at QoS requirement ,so QoS specified according to services provide across application.

2-5- QoS Verification Purposes Classification:

The QoS evaluation process is common activity, sometime the evaluation carried out to decide if the application fulfills the required QoS, this is a type of research published at this area, while the verification do to decide what the optimal application when there more than one application provide the same functionality. The another reason for QoS verification was needed to compose more than one application each one has required feature for last required services, so what is the optimal composition ,this decide according to QoS measurement. Below will mention research released about these three cases.

1-Application QoS evaluation:

The most common research area is how evaluate application performance or application QoS, more research published for this topic, because the QoS for application is most important to evaluate the application, if was performed the service which intended for as it should do, and if the QoS requirement metrics as it should be.

The QoS requirements are different according to application communities and service intended for it, so any research released concern about certain type of an applications QoS measurements and each one achieved this task using different technique. The studies published for achieve this task will mention below:

For example [4] (P. Calduwel Newton, L. Arockiam,2013) propose an evaluation strategy to ensure the expected QoS performance for radio services is similar to actual QoS because the main challenge faced services provider is that the QoS requirements differ from one application to another. The result from using this strategy in which use eight data transfer from two applications is that the expected QoS differ from actual and this information can be used in various issues related to each application.

In the study was initiated by [10] (José Darío Luis Delgado , Jesús Máximo Ramírez Santiago,2013) the TETRA network which designed to provide telecommunication services to public safety & security organizations there need to provide client required QoS. To assess achieved QoS to client the researcher propose key performance indicators (KPI) addition to monitor the TETRA parameters. The result from proposed solution is help operators to be aware of the QoS deliver meet the required by users. To accurate QoS prediction for web service [11] (Fei Peng, Xuewen Zeng, Haojiang Deng and Lei Liu,2016) proposed probabilistic matrix factor model, this model depend on users properties and their physical neighbors' performance, the result of experiment explain that the propose method performed better than state-of-the-art approaches.

To migration from traditional to cloud model; clients and Software as a Services (SaaS) provider need to establish Service Level Agreement (SLA) to certify the quality of service. The main aim of SaaS provider is to minimize cost and improve Customer Satisfaction Level (CSL) to achieve this aims the researchers [8](Linlin Wu, Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya,2014) propose customer driven SLA-based resource provisioning algorithms. These algorithms dynamically provision the resources for customers, so it reduces SLA violation and reducing cost.

In another study for QoS prediction of web services, the method used by [13](Chen Wu, Weiwei Qiu, Zibin Zheng, Xinyu Wang and Xiaohu Yang,2015) for QoS prediction called Collaborative Filtering (CF),this method depend on historical QoS data contributed by similar users and services. The problem of this method is the prediction is not accurate because it affected by untrustworthy users.to solve this problem, the researcher propose a novel credibility aware QoS prediction method called CAP. This method employs two phase k-means clustering to identify untrustworthy users. The result of the method provides considerable improvement on the prediction accuracy compared with other approaches.

In the modern software system late discover of defects led to less quality of system delivered to user, system high maintenance cost and negative impact on their user. The late discover may lead to disaster when application relate to business and safety, this disaster cause financial loss or human life loss. To avoid this losing, the researchers [14](Gerasimou, Simos, Tamburrelli, Giordano and Calinescu, Radu,2015) propose open source tool using probabilistic model checking to assess QoS at design stage to define models that satisfy QoS requirement of software system.

The study was published by [15](Lata Nautiyal and Preeti,2016) the certify of component of software to ensure that it conform to precise standard. The certification is process of bring quality to certain software product. For assurance of quality process for software component the authors use quality model to describe quality characteristics that will be taken into account during quality process assurance. To certify software component they propose weighted assignment technique, at this technique assign weight parameter of component according to their role in component.
To solve the problem of QoS requirement for e-learning application like video on demand, video conferencing, files transfer and virtual laboratory over Wifi-based Long Distance (WiLD), the authors[16](Md. I. Hussain, and N. Ahmed,2016) analyzed application using simulation. They simulate network topology and application running on network and then evaluate the parameter of QoS for each application. The result of evaluation is that the more effort required achieving good performance for e-learning application to running on WiLD network.

The task of resource management and scheduling for customers in cloud computing is complex while delivering QoS. The problem is how to simplify this task. In the past many auto scaling policies have been proposed to simplify the resource management in the work published by [56](Alexey Ilyushkin, Ahmed Ali-Eldin, Nikolas Herbst,Alessandro V. apadopoulos, Bogdan Ghit,, Dick Epema, and Alexandru Iosup,2017) [56](Alexey Ilyushkin, Ahmed Ali-Eldin, Nikolas Herbst,Alessandro V. apadopoulos, Bogdan Ghit,, Dick Epema, and Alexandru Iosup,2017) the authors proposed experimental performance evaluation of auto scaling policies. The result of study understands the state-of-the-art policy.

The wireless environment was added to Software Defined Network (SDN) so to achieve more deterministic network behavior QoS provisioning is necessary consideration. The authors[21](Rafid Mustafiz, Abu Sayem Mohammad Delowar Hossain, Nazrul Islam, and Mohammad Motiur Rahman,2017) applying the spanning tree protocol (STP) on SDWN and analyze the QoS using Mininet-Wifi STP used to observe performance of QoS parameters(bandwidth, packet transmission rate , round tripe time…etc).

In the systems deployed in the internet the QoS requirement interpolation is substantial, these systems are varying on workload, so to avoid waste of resources, the resources is allocated according to load dynamically. This system called self-adaptive systems.to analysis QoS for these systems the researchers [57](Diego Perez-Palacin, Raffaela Mirandola, and Jos´e Merseguer,2017) use accurate modeling known as Markov model under burstiness workload.

The performance evaluation of Voip for mobile user and how the QoS parameter vary for different speed. To deal with the variety the researchers [22](Ababakr Ibrahim Rasul, Diyar Salah Fadhil

and Younus Ameen Muhammed,2017) use simulation and emulation method for validity. The simulation and emulation must be match closely.

To high accurate and efficient QoS prediction for web services the authors of study [23](Jian-Long XU, and Chang-Sheng ZHU,2017) propose personalized and accurate QoS approach namely PAOMF. The model build using matrix factorization online stochastic gradient descent algorithm. The experiment apply on real dataset demonstrate that the proposed approach is effective and efficient.

In IoT application the key challenge is how to predict QoS while this application is widespread. [27](Gary White, Andrei Palade, Christian Cabrera,and Siobh´an Clarke,2017) They solve this problem using collaborative framework and the Matrix Factorization (MF) to accurately predict QoS.

In internet of thing (IoT) more data-intensive ,delay sensitive and real time application are expected to merge, so there need to ensuring QoS in terms of bandwidth and low latency, fog computing has seen as enablers for satisfying QoS requirement. The main problem which studied by[40](Ashkan Yousefpour, Ashish Patil, Genya Ishigaki, Jason P. Jue, Inwoong Kimy, Xi Wangy, Hakki C. Cankaya, Qiong Zhangy and Weisheng Xie,2018) work is dynamic fog services provisioning, it mean how deploy and release services in fog computing dynamically. to solve this problem the authors propose two heuristics, they are evaluated using simulation based on the real world traffic traces and mobile augment reality as the IoT application. The result obtained from simulation is achieving the required quality, minimizing delay and violation of SLA.

2-QoS measurement to select the optimal service:

With increase the number of services which has same functionality, regardless of the services community, the question is what criteria will help the user to choose between all available services for the required function, and how choose the service which satisfy their requirements, more research published about this problem, the researcher use QoS metrics of services as criteria to differentiation between the services and choose the optimal one from large number of application intended to present services, here the researcher need two type of techniques, one to measure the QoS for applications has same functionality, and other one used to choose between these services. The studies published for this reason are:

Cloud system is the solution for admission software products of the companies. At this field there more cloud provider to admission the software product to companies, the problem is how companies select whose satisfy their QoS requirement. To solve this problem the authors [12](Amid Khatibi Bardsiri and Seyyed Mohsen Hashemi,2014) suggest various QoS metrics for services provider to evaluate QoS for provider and select the best. The metrics associated with services provider named performance, economics and security.

The same problem in above study available on select suitable web services provider or application, but here the authors [3](Daniel G. Canton-Puerto, Francisco Moo-Mena, and Víctor Uc-Cetina ,2015) propose another mechanism to select between web services providers or applications called Hidden Markov models are probabilistic methods allow to predict behavior of web services in the near future. Also in field of selecting between cloud providers, the author [17](Pengcheng Zhang, Qing Han, Wenrui Li, Hareton Leung,and Wei Song,2016) propose prediction of cloud services QoS approach which address limitation on old prediction approaches these limitations represented on the three-layer structure on the influence of the cloud service QoS(The CPU usage, physical memory usage and the number of processes of infrastructure layer have definitely Influenced QoS) the proposed approach is Bayesian network model of QoS prediction, in this model three stages are built to prediction. the third stage is depend on experimental process using collected data from real cloud services environment. The result shows that the prediction approach is effective and accurate.

Extending on cloud services [19](Hua Ma, Haibin Zhu, Zhigang Hu, Wensheng Tang,and Pingping Dong,2017) propose a multi-valued collaborative approach for the time-aware QoS prediction of cloud services to address challenge facing consumer in selecting optimal cloud service provider (CSP) from large numbers available. These challenges representing on different QoS perspective between CSC and CSP, the long-term QoS guarantees from a CSP may not be always available and increasing in number of services user need to invoke to observe QoS for select optimal. QoS predicted via time series analysis depending on past user experiences. This approach can provide high accuracy of collaborative QoS prediction for multi-valued evaluations in the cloud computing paradigm.

The study published by [20](R. Sarala , P. Manisha, Mukku Vineesha, and G. Indumathy,2017) proposed ranking system to select web services depend on the functional relevance, user behavior, QoS and service usage factor.to accurate the result they consider consumption history, QoS preferences and QoS constraint were explored at large which is considered. The result is better in satisfying users from existing ranks system.

Continuing in field of how to fit suitable services for consumer, [58](Christos Chrysoulas, and Maria Fasli,2017) proposed QoS architecture based on set of attributes which considered when building concrete grid network for provide service.in this framework must specify QoS module that offer best QoS level, the QoS module seeking for best fit of services provider for consumer.

The authors [24](Jalel Eddine HAJLAOUI, Mohamed Nazih OMRI, and Djamal BENSLIMANE,) propose solution to address the problem of discovering and selecting configurable of cloud services and resources. They propose to employ variability modeling as a tool to considering relevant aspect of cloud services. Depend on modeling they propose

an approach for discover and select of infrastructure of cloud services. the selection of IaaS depend on their QoS and select the best ones.

To select web service which has the optimal QoS the [59](Wen-long ZHU, Bei-bei YIN,Si-qian GONG and Kai-Yuan CAI,2017) present an advanced a fully polynomial time approximation scheme (A2-FPT AS) to balance between precision and overhead.

In cloud computing, the nonprofessional users who have no usage experiences are becoming the key potential user, so these potential users need to help to select trustworthy services from abundant candidate, [60](MA Hua, HU Zhigang1 and CAI Meiling,2017) they propose trustworthy service selection approach integrating cloud model and interval number theory of potential users. They use QoS evaluation to comparison between the trustworthiness cloud services and then transform to interval number based on cloud model to create set of trustworthy services, and then use ranking system.

In mobile edge computing, the service recommendation system used to invoke service which satisfy user QoS requirement, so there need to specify QoS for services. To solve this problem [25](Shangguang Wang, Yali Zhao, Lin Huang, Jinliang Xu1,and Ching-Hsien Hsu,) propose QoS prediction taken into account mobility of mobile edge because the mobility affected on the QoS prediction. The approach used based on collaborative filtering.

Cloud applications built on service oriented architecture (SOA) to guarantee the adaptation to cloud environment chamging.so there need to guarantee QoS after changing cloud environment, the author [26](ieming Zhu, Pinjia He, Zibin Zheng, and Michael R. Lyu, 2017) propose online QoS prediction to accurately adapt the service to change with guarantee of QoS. They propose adaptive matrix factorization (AMF) to achieve the accuracy.

In mobile service, with the increase number of services has the same functionality need to select a mobile service from services of candidate service. The select operation depend on service QoS , but in mobile service large part of QoS values are unknown , to solve this problem [61](Yuyu Yin, Wenting Xu, Yueshen Xu, He Li, and Lifeng Yu,2017) propose combination between two technique to solve all issue related. They use filtering base collaborative filtering extend slope one model (FB-CF) and filtering base and matrix factorization (FB-MF).

To select services from multi services has same functionality, [30](Ireneusz Jozwiak, Michal Kedziora and Aleksander Marianski ,2018) they use modified Analytic Hierarchical Process (AHP) combined with obtaining probability distribution function of QoS parameter.

3-QoS measurement to optimal composition:

When the user requirements cannot satisfy using one service and the requirements of user available at deferent services, the need for composite more than one services was arises, the composition will solve the problem of user satisfaction, but what is best services composition is optimal for user requirements. The criteria used to differentiate between compositions solutions is QoS parameters measurements. The studies published for this reason are:

When manufacture cloud service composition, must consider the ability of correlation among different manufacturing cloud service. With similar function but different QoS the authors of the paper [31](Hong Ji, Xifan Yao , and Yong Chen,2015) present correlation –aware manufacturing cloud services description model to distinguish QoS dependence of an individual services on other related services. They propose approach based on genetic algorithm correlation-aware optimal service selection. The result of the services composition on higher quality can be obtained when correlation was considered.

In web services when user needs optimal solution or single service cannot satisfy their requirement or more than one service provide the same functionality, at these cases there need to composition more web services to fulfill user requirement. [32](Kirit J. Modi, and Sanjay Garg,2015) present dynamic composition web services depend on QoS parameter to select the most relevant using genetic algorithm.

The study published by[33](Pablo Rodriguez-Mier, Manuel Mucientes, and Manuel Lama,2015) addressed the problem of composition when there large amount of possible composition depending on services functionality and QoS. They propose hybrid approach for automatic composition of web services based on optimal end-to-end-QoS minimizing the number of services of the resulting composition. The resulting of using approach is that; perform better than the state-of-the-art Obtaining solution with less services and optimal QoS.

The authors [35](Yan Guo, Shangguang Wang, Kok-seng Wong and Myung Ho Kim,2016) propose service selection approach based on QoS prediction to composition optimal services. To predict QoS for web service they consider historical QoS information as a time series and predict QoS values using the autoregressive integrated moving average model to provide more accurate QoS attribute values.

In the study published by [34](Amina BEKKOUCHE, Sidi Mohammed BENSLIMANE, Marianne HUCHARD, Chouki TIBERMACINE, Fethallah, HADJILA, and Mohammed MERZOUG,) they propose solution to composition problem focused on three dimension: (1) fully functional (i.e. fully executable) by using a mechanism of semantic matching between the services involved in the solutions, (2) are optimized according to non-functional Quality of Service (QoS) measurements, and (3) respect global QoS constraints. They propose novel approach based on harmony search (HS) algorithm. The result shows that, approach is efficient and effective to extract the optimal or near optimal composition

in diverse scenarios; and variants HS algorithms have brought improvements in terms of fitness and execution time.

The author [62](Soumi Chattopadhyay and Ansuman Banerjee,2017) propose solution for balancing the tradeoff between compute efficiency and optimality in service composition, the optimality mean the composition services satisfaction of QoS required by user. They use abstraction refinement methods which give speedup compared to traditional composition techniques.

The problem of finding optimal QoS available services for composition addressed by [38](Samia Sadouki Chibani and Abdelkamel Tari,2017), it consider is optimization problem, they present meta-heuristic bio-inspired to QoS-aware web service composition. It based on elephant herding optimization (EHO) algorithm. The evaluation of proposed solution explains that it better than existing algorithm.

In services composition when large scale of services needs to search in to find the optimal services according to user requirement, the problem is time consuming. To solve this problem [63](Sepideh Sheivandi, and Sima Emadi,2017) they use method based on modified algorithm of graph coloring. They implement MGC-TOP K and MGC-K.

To provide wide range of composite cloud services providers need to establish mutual agreements, so providers can compose effective and efficient service workflow, the authors [28](Fabrizio Messina ,Giuseppe Pappalardo, Antonello Comi, Lidia Fotia, Domenico Rosaci and Giuseppe M.L. Sarné**,**2017) propose reputation model to support the composition of of complex cloud services considering cost and measure QoS. The model evaluated by set of experiments.

[29](Chandrashekar Jatoth , G.R. Gangadharan, Ugo Fiore, and Rajkumar Buyya,2017) they address the problem of composition by reducing optimal QoS that satisfy the customer requirement in big services environment. They propose novel MapReduce based evolutionary algorithm with guided mutation considering QoS.

The evaluation of proposed mechanism that is outperforms other methods. the researchers [39](Elsoon Neshati and Ali Asghar Pourhaji Kazem,2018) using an ant colony optimization algorithm to composition cloud manufacturing services considering quality of services. They apply a novel ACO algorithm to QoS-aware manufacturing cloud service composition problem. The simulation used to evaluate the approach explains that; the ACO algorithm has good convergence speed and stability.

2-6-Analysis of published research:

1-QoS Aspects Which Considered by Researchers:

To evaluate QoS for applications or network or any services presented to customers, there two types of measurements, these types are quantitative (ex: bandwidth, delay and jitter) and qualitative (ex: reliability and security), and may called functional and nonfunctional QoS, each one has parameters used to evaluate the QoS. According to applications or purpose of evaluation specify the parameters used in the process of evaluation. Most of studies concern about measure response time and throughput because they are web application or cloud services addition to another parameters; these parameters also used in evaluation of QoS is reliability, availability, and in case of cloud services the security is a most important parameters. The twenty nine studies from thirty nine use response time and throughput as indicator to QoS

2-Comporison between techniques used to measure QoS:

Here will mention the techniques used to measure or predict QoS and comparison between these techniques to show what is effective and produce accurate value, in table 2.1 demonstrate each technique and number of studies used it:

Probabilistic model used to build behavior models, these models used to predict behavior in near future. Depending on this prediction can also predict QoS of system, so the QoS prediction depends on probabilities used in building of behavior model, [3] and [14] use this techniques to benefits from its ability in prediction. Another technique used in evaluating QoS is strategies; the strategies are dividing the evaluation of QoS process to steps to reach to values. The steps begin from specify the standard values of QoS to monitor the real values and then comparison between standard values and monitored values to classify the QoS for the system. The strategies method is effective in classifying performance of system. This technique used by researcher [10].

[8] The researchers not use QoS evaluation technique but use algorithm to provide required QoS resources for system. The algorithm depends on (SLA) Service Level Agreements and (CSL) Customer Satisfaction Level. The algorithm is more effective in provisioning resources. In study [12] the author use metrics to evaluate QoS, the metrics is features of services need to be evaluated, and this metrics consider as standard for assessing service, this is good in evaluating service in comparison services.

[13] the author use key-mean clustering to identify untrusted data, because this data used in predicting QoS, so must use trust data, this technique is type of unsupervised learning which is used when you have unlabeled data (data without defined categories or groups) the good of this algorithm is to fined groups in the data, with the number of groups represented by the variable k. clustering allows you to find and analyze the groups that have form organically. According to previous definition, this technique was effective in groupie the data and then use clustering to analyze these data which used in predicting QoS.

Another technique used by researcher [15] is unstructured weighting technique; this technique is used to assign weight to standard factors used in assessing the quality of software. The weight is given depending on common understanding of the system and their experience, so this technique is imprecise hundred percent. Consequently can say this technique is un effective.

In [16][21][22] the author use simulation software and emulator tools to represent the real environment which need to analyze his Quality, and then observe the quality parameters. These mechanism consider more effective because the results was given specify what the component need to improve or what is cause the degradation and the plan to develop.

Another technique used in prediction QoS is Bayesian Network, this technique belong to family of probabilistic graphical models (GM), it is used to represent knowledge about uncertain domain, the node of graph represent variable while the edge between nodes represent probabilistic dependencies among variables. This technique used by researchers in [17] to predict QoS for cloud computing because they take into consideration the characteristic of cloud computing itself (software and hardware) so this technique is most effective in predict QoS.

The researchers in[18] use technique Autoregressive Integrated Moving Average (ARIMA) this technique is time serial analysis model which is generalize of ARMA model, this two models are fitted to time series data either to better understand or to predict future points in series it applied in some case where data show evidence of nonstationary.it is inflexible and not accurate.

One of technique used in predict QoS is Collaborative Filtering (CF), it is a way of making automatic prediction (filtering) about the interest of user by collecting preference from many other users (collaborative) the author of [19] and [25] use CF based in neighboring users, they use historical data to predict QoS, so they finding similar users and services and mining their similarities and calculate unknown data of similar users or services. Consequently CF is most effective in calculating QoS.

Weighted Additive is strategy used calculate QoS score, is found the summation of the product of each QoS constraint with the QoS weight obtained from correspond QoS preference specified. This strategy used by author of [20].

The author of [11],[23],[26]and [27] use technique in predicting QoS, this technique is matrix factorization MF, also called matrix decomposition is factorization of matrix in to a product of matrices.is dimensionality reduction technique, it used to predict QoS from matrix constructed from users and services, if there services un run from matrix using matrix factorization to predict this un use service.th MF is more accurate technique in predicting QoS because can use any factor effect on QoS in predicting using matrix.

The author of [28] use reputation feedback and measure system to evaluate services, this method not effective because depend on user opinion.

The technique called MapReduce is weight of QoS attributes, is user preference obtained by Analytical Hierarchy Process (AHP), AHP is method to derive ratio scale from paired comparison, hence the input weight of QoS attribute. This technique used by author of [29] and [30]. Directed weighted graph technique used by author of [24], is graph has nodes (vertex) and edges, the properties represented using edges, assign weight to represent quality attribute value, the graph can transfer to matrix. Table 2.1 explains each technique and studies which use this technique and number of studies.

Table2.1 statistics of techniques and No. of studies use them.

No	Technique Name	Number of Studies	References
1.	Probabilistic model	2	[3][14]
2.	Strategy	1	[10]
3.	Provisioning Algorithm	1	[8]
4.	Metrics	1	[12]
5.	Key-Mean clustering	1	[13]
6.	Unstructured Weighted	1	[15]
7.	Simulation and emulation	3	[16] [21] [22]
8.	Bayesian Network	1	[17]
9.	ARIMA	1	[18]
10.	CF	2	[19][25]
11.	Weighted Additive	1	[20]
12.	MF	4	[11][23][26][27]
13.	Reputation	1	[28]
14.	Weighted techniques	3	[24][29][30]

The table shows that the most technique used is matrix factorization, this technique is used across deferent years, the first paper consider at this review released in 2014, and the last paper released in 2017, that lead to substantiation of efficiency of this technique.

3-Researches Contributions:

Each study has its own contribution produced from excusing proposed solution, according to main contributions of studies can classify these studies to five classes. Table 2.2explain the contribution and number of studies which produce the contribution, observe that, the most of studies contributions is a model.

Table2.2 Studies Contributions

Contribution	No of studies	References
Model	15	[3,11,8,12,13,15,17,28,31,32,33,34,35,36,37]
Framework	3	[23,26,28]
Approach	5	[19,25,38,39,40]
Strategy	1	[10]
System	1	[20]

2-7-Summary:

The QoS is greate research area because the defect in any kind of systems sometimes leads to disaster. More researches available about the QoS, the QoS metrics deferent from one service to another because the QoS parameters deferent according to a service which need to evaluate, therefore any researcher concern about certain parameters, here was summarized numbers of researches published about QoS, and demonstrated each study and problem side concern about, and technique used for proposed solution, also I was comprise between these technique, finally classify the studies according to contributions .

CHAPTER III

Methodology and Research Framework

3-1-Introduction:

The problem must solved through this research is validity of QoS for application, it mean test if the application after running perform the task according to required QoS which specify according to application community. The validity process will perform for QoS parameters. The case study will take into account is web application, so the requirement specify as standard of web application and the user needs.

Because the QoS is not known, so the question here is how measure and validate unknown values. To treat with the all sides of this problem, we will use probabilistic techniques, according to ability of these techniques will represent QoS requirements of application in term of probability, these probabilities will represented in system after transfer it to processes (states) using Markov chains, and then use probabilistic model checking to validate QoS probability. According to systems time; discrete time Markov chain or continuous time Markov chain, states properties choose technique to represent these properties as probabilities. Here we will talk about these techniques in addition to Markov chains in more details, and how will use in research.

3-2-The Proposed Solution Hypotheses:

The application type used as a case study is a web application, this type of application has standard requirements which is general for all, addition to requirements specify according to more factors: service which the application developed for, the environment the application running in, and user experience. The proposed solution for the validity of QoS is probabilistic model checking. The proposed solution has a number of stages must excuse to gain result of validity and then decide if the application achieve user satisfaction or not, at each stage use technique to achieve it, in below section will talk about techniques and stages used in.

Steps to verify of QoS for application:

1. Specify case study (system).
2. Describe system processes.
3. Specify QoS requirements for the system and assign the values of QoS requirements.
4. Represent the system models using DTMC and CTMC according to requirements.
5. Transfer the models to module using PRISM modeling language.
6. Specify properties using PCTL or CSL and add it as property list in PRISM.
7. Run the module in PRISM and analysis the result.

The mentioned steps clear on figure 3-1

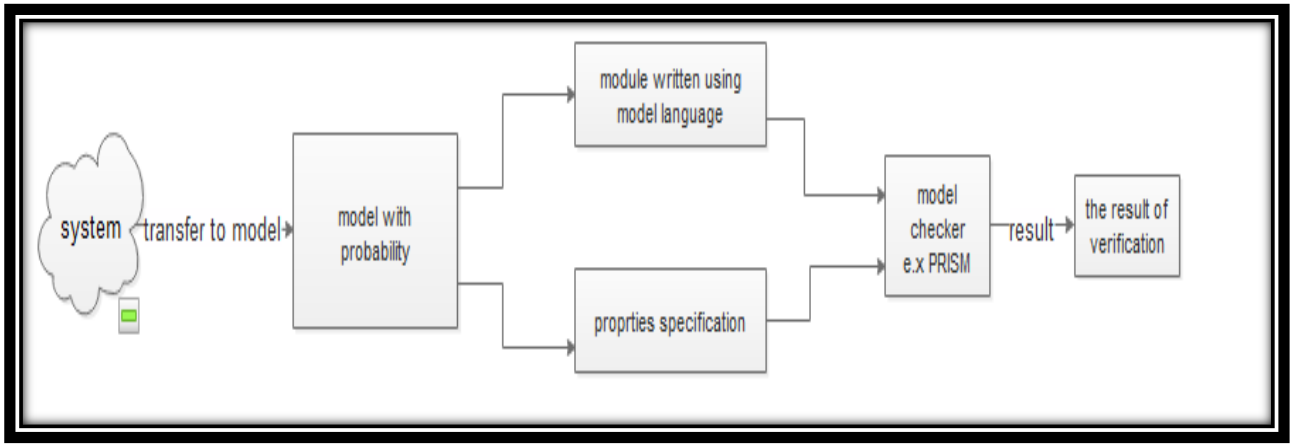


Figure 3.1 model checking steps

3-3-Research Framework Description:

To solve the validity of QoS problem, certain processes will follow according to proposed solution; to test proposed solution and evaluate the accuracy of QoS will use two case studies, and represent the solution as stages to gain results. Below describe the solution framework.

1-Specify Case Study:

To verify proposed solution I use two case studies, these case studies are web application, one of them is small system for procurement, another one for department of commercial registrar, is a department at ministry of justice, and the second system has multi processes, the details of description for every system and all states of systems will be clear on the research implementation.

2-Specify Systems Requirements:

The requirements of system are critical point at this research, because the validity will be verified for requirements. The requirements specified according to the properties which need to validate, these properties are the QoS parameters, so the validity of requirements means QoS evaluations.

According to proposed solution the requirements representing as probability values after represent systems states transitions using markov chains, the total of probabilities of transition from state to anther must equal to one.

3-Formal Representation for QoS Requirements:

To represent QoS requirements, two techniques available, each one use according to system state transition time; PCTL this for represent discrete time, and CSL for represent continuous time, at this step the action is clarify formula for each on.

4-Quality Evaluation Model:

At this stage represent system as it run in real life; system states and their transition, also there two model one for discrete time and other one for continuous time.

5-write Algorithm:

The algorithm will write to show the paths of execution of evaluation model, if the event is discrete running module deferent from continues event.

6- Execution Evaluation Model:

This is main stage for the proposed solution, can divide to multi point:

i. Represent the system processes using Markov chains to clarify the system states and probabilities of transitions between these states. Need to two diagram (discrete and continuous time).
ii. Represent the system requirements as probability values.
iii. From diagram of represent system states can write module using model language to run in model checker software, here will use PRISM model checker.
iv. Run the module to gain result for analysis the system performance.

3-4-Summary:

The proposed solution for the problem of validity of QoS is a formal verification technique used to modeling and analyzing systems to verify if the system will satisfy user requirements, this technique is probabilistic model checking, then to gain results from the model will use software called PRISM, the results gained from the model checker will use to make recommendations and decision about the system which is use as a case study.

CHAPTER V

Implementation and Validate QoS Requirements

4-1- Introduction:

In this chapter presents the results of execution of evaluation model which mentioned in previous chapter and then analysis the results to list of finding from the experiment. Before mention the results will demonstrate the execution of model in details.

4-2-Overview of Framework of Implementation:

The research assumption implemented in two case studies, they are web applications, and each one has different requirements, therefore different QoS matrices and evaluation models. After specified requirements for each application, transferred the application processes (states) which need validity to marckov chains contain nodes and vertices, the nodes represent states and the vertices represents the transition between states, the transition represent the probability of next state, the total of probability from on state equal to one. From models specified properties which used to validity of QoS, these properties represented using PCTL and CSL according to states time required. Then to validate transferred the model using model language to module which run on PRISM, and properties written as property list using property specification.

The properties are representing the factors effect on system QoS, and which need to validate. After modules was written and their associated property list added, the module be ready to run and obtained the results of execution to analysis.

4-3-Proposed Solution implementation:

The description of implementation was described earlier in chapter three, here will implement the proposed solution to gain and analyze results to evaluate the proposed solution. The solution model will implemented for two systems as case study.

1-The case study No.1:

1. **Description of system:**

The first case study is the Procurement system; is a web base system owner by ministry of justice, is having three layers as in figure 4.1:

1. Client layer.
2. Application layer.
3. Database layer.

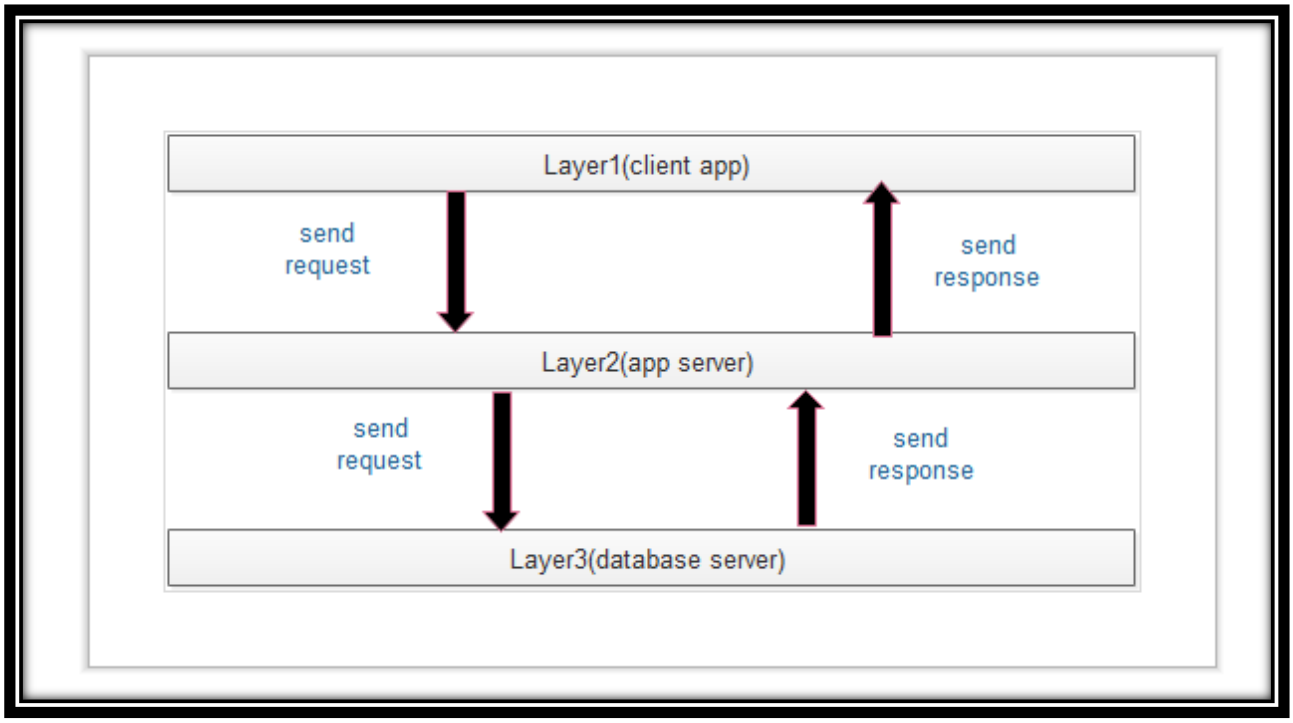


Figure 4.1 Procurement system layers.

The application layer is application host on leased server and stay between client application and database server, also database stored on leased server for increase security. Application server receive request from client application and send it to database server, the request is a query may contain insert, retrieve, or validate data, and then inverse receive result from database server and send it to client application to show the result to user. This pathway is applied also in store data in the database. Here will verify the system security representing in session timeout and lock users if will try the login operation and failed to login at certain trial times. Also will verify system availability, to achieve this task will test the connectivity depend on probability of failure.

The description of system processes:

1. The system processes from login screen to main menu.
2. The main menu consists of all process of system (see figure 4.2).
    a. Order: the process of add new order and assign key.
    b. Bills: any order must have 3 bills maximum after addition three bills will transfer from new order.
    c. Schedule : any order served according to their priority
    d. Sort: sort the bill which has fewer prices.
    e. Budget : according to the budget available choose the high priority order
    f. Search
    g. Report

h. Ratification: at this process print the ratification of certain order to manually transfer to accounting unit.
i. Payments: assign number of ratification to the order which ratification.
j. Indebtedness
k. Archives: all order ratification must transfer to archiving

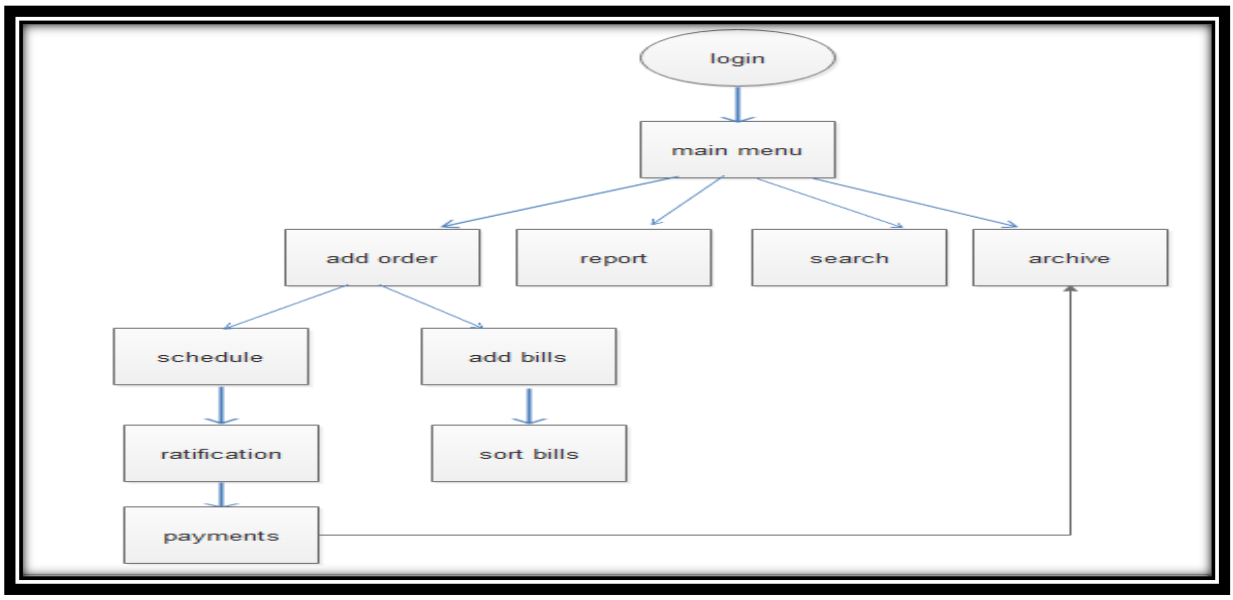


Figure 4.2 procurement systems Flowchart

2. Propose and Formal Definition of QoS requirement for application specified :

The requirements representing in probability form according to system states as follow:

i. Security requirements:
   1. R_1: The probability P_1 of lock user account if he try more than 3 times $P_{1=}$ 0.02.
   2. R_2: The probability P_2 of user log out if he take time of rest without transaction more than 60 seconds $P_2 \leq 0.5$.

ii. Process execution time requirement:
   1. R_3: The probability P_3 of request of site failed if it takes more than 120 seconds $P_3 \leq 0.3$.
   2. R_4: The probability P_4 of any process will drop if it takes more than 10 seconds $P_4 \leq 0.2$.

iii. Availability requirements:

1. R_5: The probability P_5 of connection for application server failed P_5 =0.1.
2. R_6: The probability P_6 of connection for database server failed P_6=0.02.

After specify QoS requirements for system can use formal language to formal definition of QoS requirements, use probabilistic computation tree logic (PCTL) and continuous stochastic logic. The syntax of PCTL as follow:

States formula: $\Phi ::= true|a|\ \Phi \wedge \Phi\ |\ \neg \Phi|\ \rho \bowtie p\ [\Psi]$

Path formula: $\Psi ::= X\ \Phi|\ \Phi U^{\leq k}\ \Phi$

K is steps to reach states

Where a is an atomic proposition, $\bowtie \in \{\leq,<,\geq,>\}$ and $p \in \{0,1\}$

X operator called next state and U is until operator. The formula X Φ is true for a path $\omega \in path_s$ if Φ is satisfied in the next state, and $\Phi_1 U \Phi_2$ is true if Φ_2 is satisfied at some state along the path and Φ_1 is true up until that point.
The syntax of CSL as follow:

States formula: $\Phi ::= true|a|\ \Phi \wedge \Phi\ |\ \neg \Phi|\ S \bowtie_p [\Phi]|\ \rho \bowtie p\ [\Psi]$

Path formula: $\Psi ::= X^{\leq t}\ \Phi|\ \Phi U^{\leq t}\ \Phi$

Where a is the atomic propositions, $\bowtie \in \{\leq,<,\geq,>\}$, $p \in [0, 1]$, and $t \in R \geq 0$.
Steady state $S \bowtie_p [\Phi]$ is probability that Φ holds in steady state is $\bowtie_p$, $\rho \bowtie p\ [\Psi]$ the probability that path fulfill Ψ is $\bowtie p$.

3. Quality Evaluation Model:

a. Discrete model:

Path-based properties of discrete model shaped using state-transition systems compound with probabilities of model states S.in our model how to constructing the discrete time and distribution and transition matrix.

-The Transient state distribution: $\underline{\pi}s,k$

-vector $\pi s,k$ i.e. $\pi s,k(s')$ for all states s'.

- The transient state probability: $\pi s,k(s')$.

b. Continuous model:

Transitions between states can occur at any (real-valued) time instant

-State of the model at a particular time instant

-$\pi^{c}s,t(s')$ is probability of, having started in state s, being instate s' at time t (in CTMC C)

- $\pi^{c}s,t(s') = Prs\{ \omega \in Path^{C}(s) \mid \omega@t=s' \}$

4. QoS Validity Algorithm:

To validate QoS for procurement system we used an algorithm which uses the requirements of system as inputs(R) and outputs will be is result of validity of QoS requirement (VR) for system by use model checker. Since there are two models, the state can be from one of model DT or CT, and transition between states is the probability and in case of CT there need to specify time of state. The algorithm is listing below:

Input: R QoS requirement, S states for two models, T transition between states (probability) and ST is time required for state in case of CT.

Output: VR_{DT} validity of QoS for discrete time DT, VR_{CT} validity of QoS for continuous time CT.

For each $r \in R$

if ST ≠ continuous then

//Represent by PCTL

S→ Φ::= true |a| ΦΛ Φ| ¬ Φ | $\rho_{\bowtie p}$ (Ψ);

T → Ψ::= X Φ | Φ $U^{\leq t}$ Φ;

Check the validity(R);

Return VR_{DT};

ELSE

//Represent by CSL

S→ Φ::= true |a| ΦΛ Φ| ¬ Φ | $S_{\bowtie p}$| $\rho_{\bowtie p}$ (Ψ) ;

T → Ψ::= $X^{\leq t}$ Φ| Φ $U^{\leq t}$ Φ;

Return VR_{CT};

End if;

End each;

5. Execution Evaluation Model:

According to time events can decide if will use DTMC or CTMC to represent QoS requirements in feature of Markov chains, firstly represent requirements in discrete formal,

and figure 4.3 explain QoS requirements representation, and then represent requirements in continuous formal, as figure 4.4

The requirements R_1 and from R_5 to R_6 representing in form of DTMC

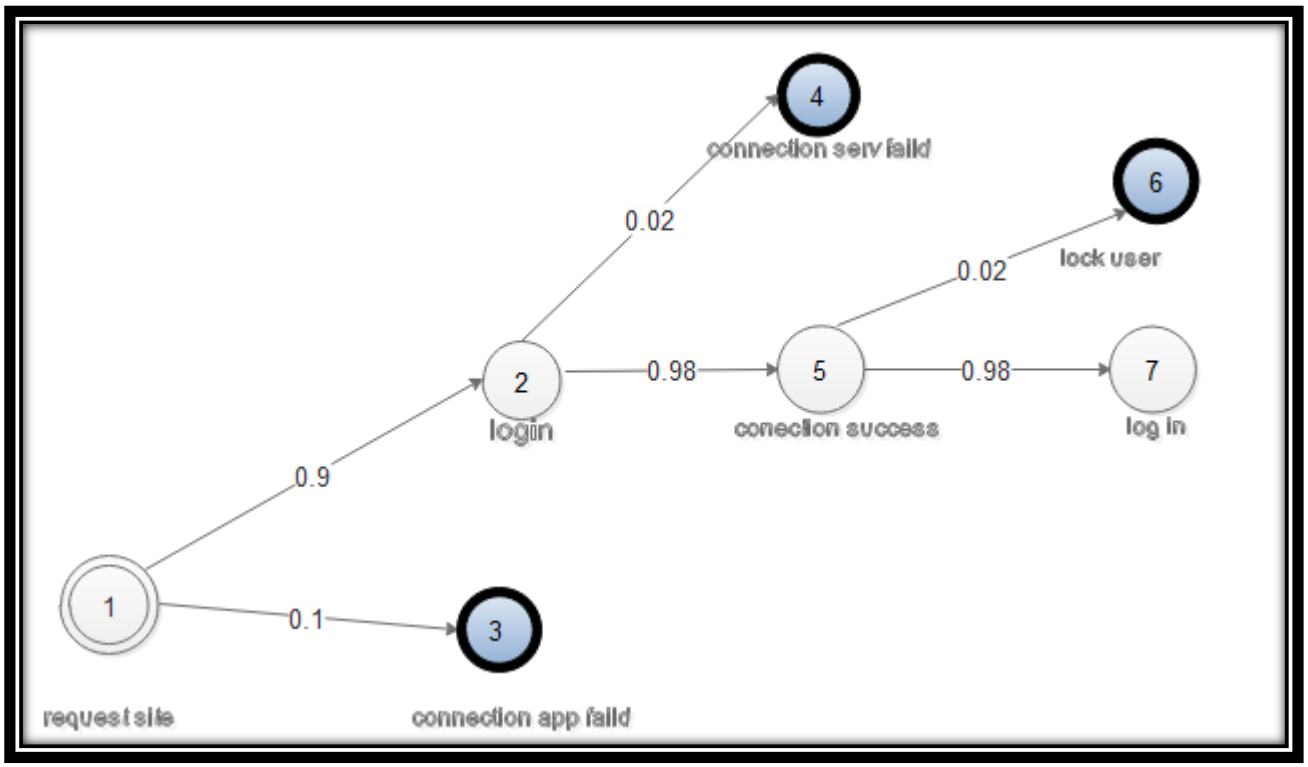


Figure 4.3 Procurement DTMC

The requirements can be translated to PCTL as follow:

$P_{\geq 0.1} = [\diamond\, s = 3]$ The probability of reaching state called connection application failed is greater than or equal 0.1.
$P_{\geq 0.02} = [\diamond\, s = 4]$ The probability of reaching state called connection serv failed is greater than or equal 0.02.
$P_{\geq 0.02} = [\diamond\, s = 6]$ The probability of reaching state called lock of user is greater than or equal 0.02.

The requirements from R_2 to R_4 representing in form of Continuous Time Markov Chain.

Here will represent state rate time

Table 4.1 state rate times

	State rate	Value req\sec
1.	Login	120
2.	Main menu	20
3.	Select process	10
4.	Success process	10

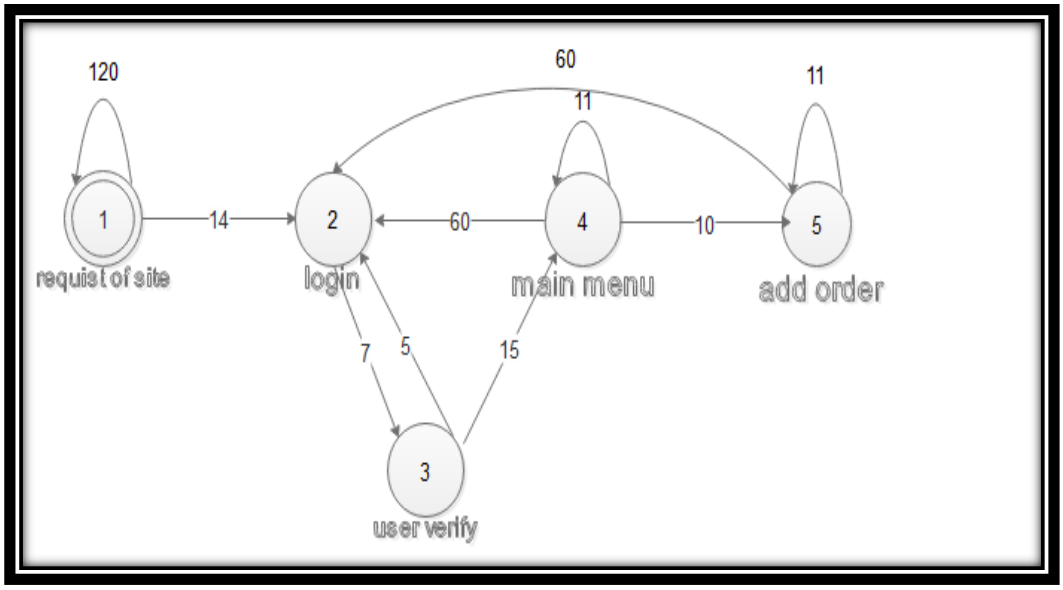


Figure 4.4 procurement CTMC

The requirements can be translated to CSL as follow:

$P_{\leq 0.5} = [\diamond^{\leq 60}\ s = 2]$ The probability of system back to login state within 60 s is less than or equal 0.5

$P_{\leq 0.3} = [\diamond^{\leq 120}\ s = 1]$ The probability of system stay at request of site within 120 s is less than or equal 0.3

$P_{\leq 0.2} = [\diamond^{\leq 10}\ s = 5]$ The probability of system drop any of process and back to previous within 10s less than is less than or equal 0.2.

After represented QoS requirements using Markov chain, the next step on model checking is transfer the states and navigation probabilities using model language to model checker software PRISM to check the model properties. To verify the QoS requirements write two modules depending on Markov chain system state representations

1. Representing DTMC in PRISM model checker which act security and availability requirements:

```
dtmc

module DTMC_case1

 ds:[1..7]init 1;

 fs:[3..6] init 5;

 []ds=1->0.1:(ds'=3)+0.9:(ds'=2)&(fs'=3);

 []ds=2->0.98:(ds'=5)+0.02:(ds'=4)&(fs'=4);

 []ds=5->0.02:(ds'=6)&(fs'=6)+0.98:(ds'=7);

 endmodule
```

The results of experiments which gained from module according to properties when use PRISM explained in table 4.2:

Table 4.2 comparisons between proposed and experiment of QoS requirements

QoS requirement	QoS aspects	QoS probability value	Experiments result
Probability of connection to application server failed	Availability requirement	0.1	0.1
Probability of connection of database server failed	Availability requirement	0.02	0.0180
Probability of user locked	Security requirement	0.02	0.01764

The experiment results explain that, the probabilities produced equal and less than probabilities supposed on system requirements, so the system satisfy user requirements.

In figure 4.5 the graph generated from run experiments explain the values of QoS requirements properties, each requirements run on different experiments and then integrate all experiments on one graph , and another graphs generated from recurrent experiment will explain in appendix A.

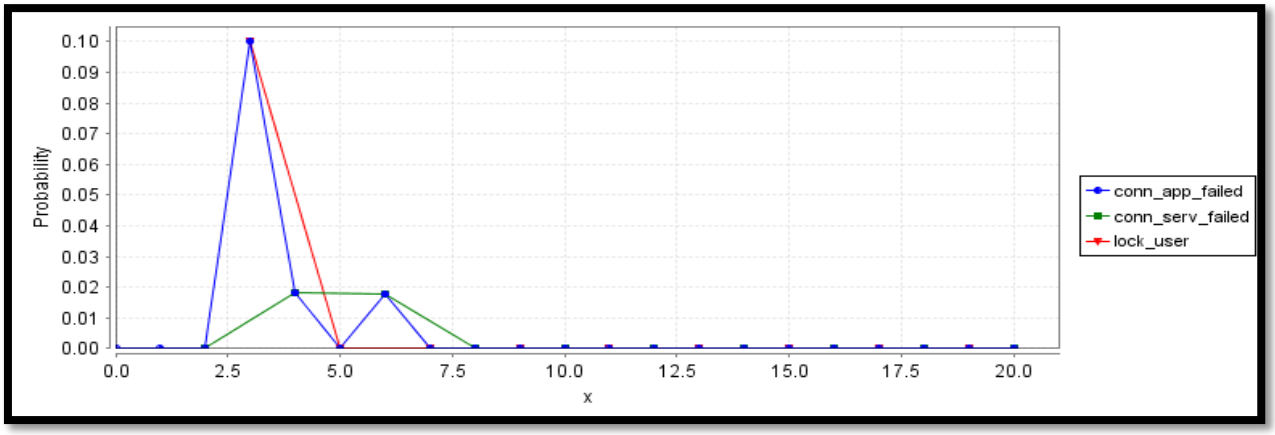


Figure 4.5 analyses of security and availability

2. Representing CTMC in PRISM model checker:

```
ctmc

module ctmccas1

        st:[1..5] init 1;

        []st=1->14:(st'=2)+120:(st'=1);

        []st=2->7:(st'=3);

        []st=3->15:(st'=4)+5:(st'=2);

        []st=4->10:(st'=5)+60:(st'=2)+11:(st'=4);

        []st=5->11:(st'=5)+60:(st'=2);

endmodule
```

The experiment result explained in table 4.3:

Table 4.3 comparisons between proposed and experiment of QoS requirements

QoS requirement name	QoS requirements aspects	QoS probability value	Experiments result
Probability of loading time out	Process execution time requirements	<=0.3	1
Probability of session time	Security	<=0.5	1

out	requirements		
Probability of drop process	Process execution time requirements	<=0.2	1

According to experiment result the system was violated, so does achieve user satisfaction because the probabilities result from experiments is greater than the probabilities supposed on QoS requirements.

In figure 4.6 the graph generated from run experiments, and another graphs generated from recurrent experiment will explain in appendix A.

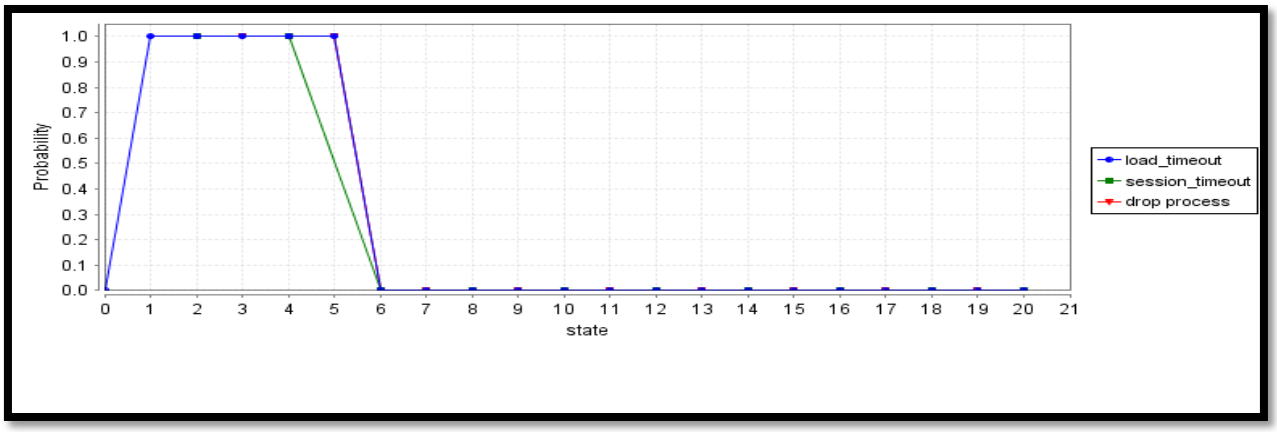


Figure 4.6 analyses of security and execution time requirement.

2-Case study No.2:

1. **Description of system:**

The second case study is commercial entities registration and follows up, is web application owner by commercial registration management, like first case study, also has the same layers, but the different between the two systems in processes and the properties of network running in.

1. The system environment:
    a. It run in LAN and access to servers across the national network.
    b. The head office has about 83 PC and able to growth.
    c. The bandwidth of network is 2MB/S
    d. There about ten offices at different states each office has number of PC between 3 and 5.
    e. The PC will verify across IP address plus MAC address.
    f. The national network don't pass any IP not configure in firewall.
2. The system process description:

a. The system start from login screen lead to main menu after verification of user.
b. Invalid user cannot log in to system
c. If user tries to log more than 3time and fail the account is blocking.
d. The user account verification is mechanism of security to save data from corrupted or stealing.
e. Each user has privileges to achieve certain process.
f. The main menu consists of all process of system:
    1. Common search.
    2. Companies.
    3. Business names.
    4. Commercial agencies
    5. Certificates print.
    6. The reports.
    7. Management of users.
    8. System management.
g. Some processes represent department processes.
h. Another process represent system management processes
i. Each department process consists of two sup department; new registration and follow of registered entities
j. Each operation of insert data to database is encrypted data before.
k. The password of each user encrypted before send to database.
l. Any update operation for database or application not allow for all just for certain IP address configure before.

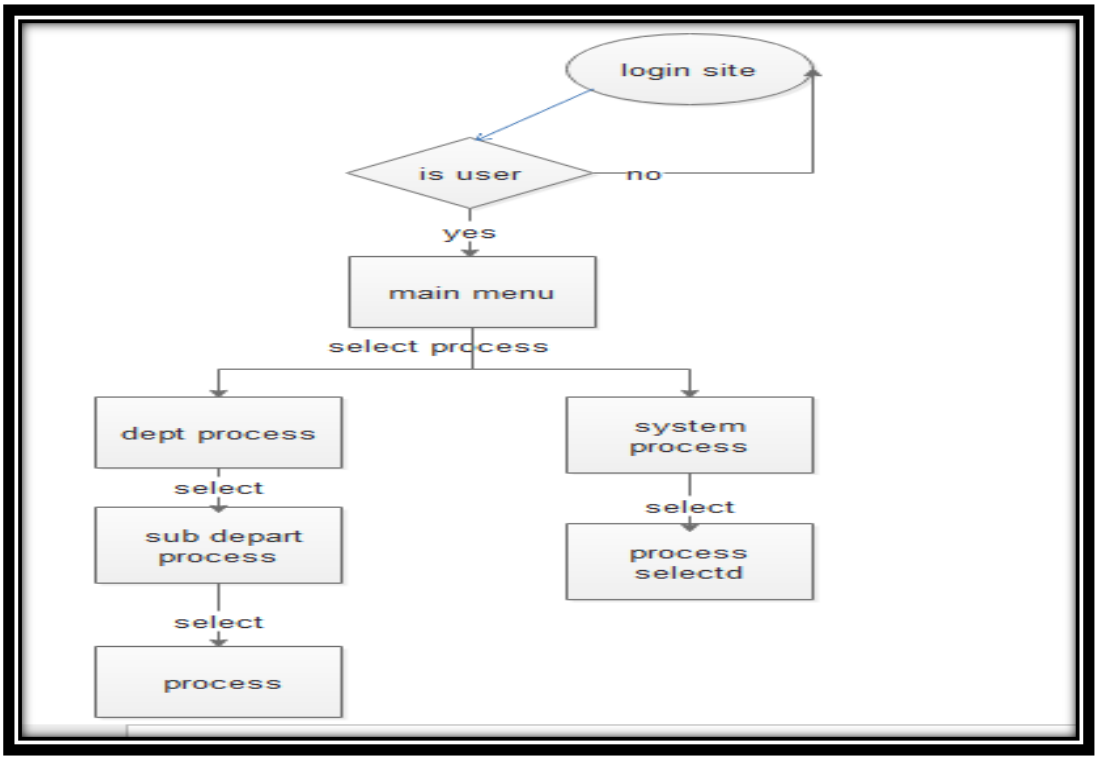


Figure 4.7 commercial registrar system flowchart

2. Propose and Formal Definition of QoS requirement for application specified:

The requirements representing in probability form according to system states:

i. Security requirement

1. R_1: The probability P_1 of lock user account if he try more than 3 times $P_{1=}$ 0.8 .
2. R_2 : The probability P_2 of account not log out if he take time of rest without transaction more than 20 seconds $P_2 \leq 0.01$.
3. R_3 : The probability P_3 of updating database or application from not allowed IP address P_3 =0.001.
4. R_4 : The probability P_4 of hacking key encryption of data P_4 = 0.0001.
5. R_5 : the probability P_5 of hacking key encryption password P_5=0.001.
6. R_6 :the probability P_6 of allow to user do unauthorized process P_6 =0.001.

ii. Failure Rate:

1. R_7 : the probability P_7 of request of site fail during high load time if it take more than 20 second P_7 = 0.05.
2. R_8 : the probability P_8 of fail of process during high load time if it take more than 20 second P_8 = 0.05.

iii. Availability Requirements:

1. R_9 : The probability P_9 of connection for application server failed P_9=0.001.
2. R_{10} :The probability P_{10} of connection for database server failed P_{10}=0.0001.

Formal Representation of QoS Requirement as mentioned in case study number one

3. Quality Evaluation Model:

a. Discrete model:

Path-based properties of discrete model shaped using state-transition systems compound with probabilities of model states S.in our model how to constructing the discrete time and distribution and transition matrix.

-The Transient state distribution: $\underline{\pi}s,k$

-vector $\pi s,k$ i.e. $\pi s,k(s')$ for all states s'.

- The transient state probability: $\pi s,k(s')$.

b. Continuous model:

Transitions between states can occur at any (real-valued) time instant

-State of the model at a particular time instant

-$\pi^c s,t(s')$ is probability of, having started in state s, being instate s' at time t (in CTMC C)

- $\pi^c s,t(s') = Prs\{ \omega \in Path^C(s) \mid \omega@t=s' \}$

4. QoS Validity Algorithm:

Will use the same algorithm in case study number one

5. Execution Evaluation Model:

Figure 4.8 representing requirements in form of Discrete Time Markov Chain , and figure 4.9 represent QoS requirements in form of CTMC:

The requirements R1,R3 to R6,R9 and R10

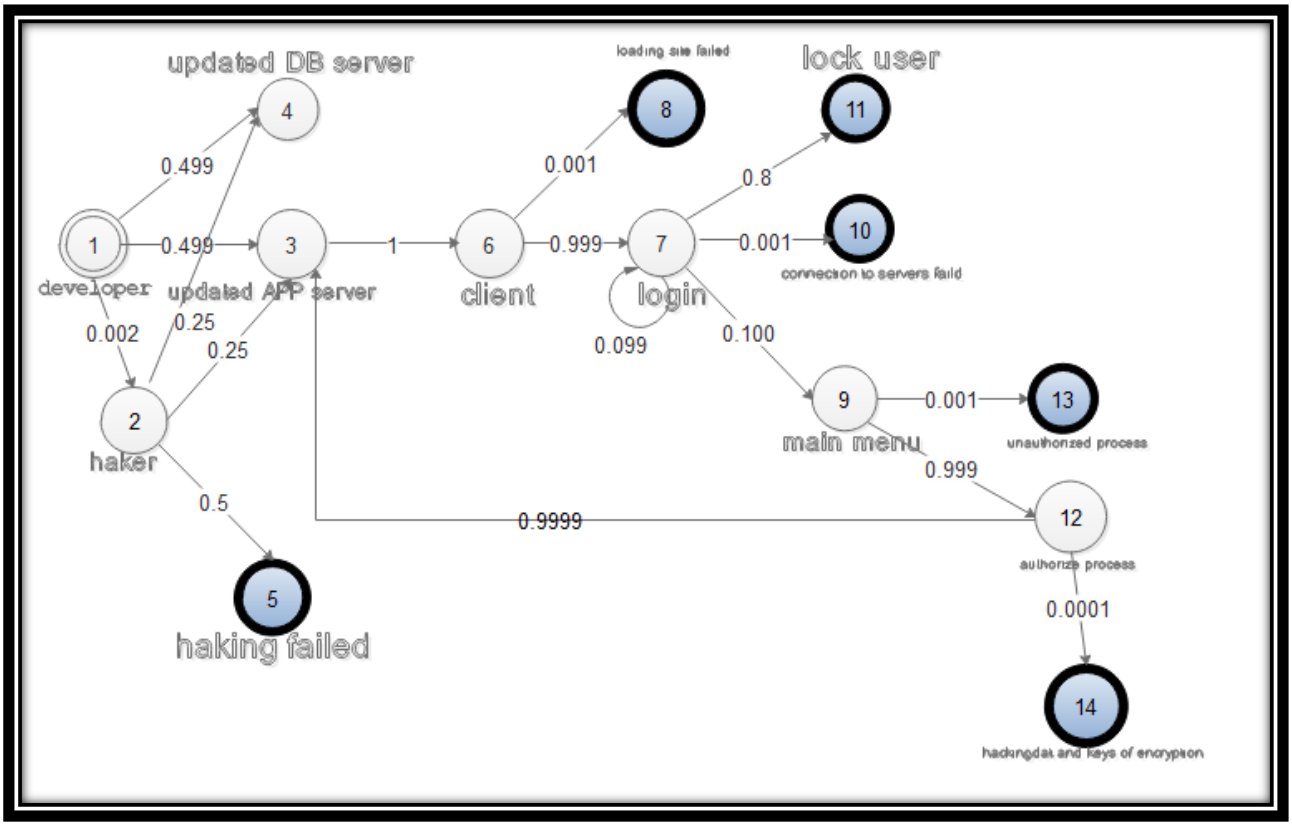


Figure 4.8 commercial registrars DTMC

The requirements can be translated to PCTL as follow:

$P_{\geq 0.5} = [\diamond\ s = 5]$ The probability of hacker failed to update DB and app server is greater than or equal 0.5.

$P_{\geq 0.001} = [\diamond\ s = 8]$ The probability loading site failed greater than or equal 0.001.

$P_{\geq 0.8} = [\diamond\ s = 11]$ The probability of user lock if do three trials are greater than or equal 0.08.

$P_{\geq 0.001} = [\diamond\ s = 10]$ The probability of user verification failed is greater than or equal 0.001.

$P_{\geq 0.001} = [\diamond\ s = 13]$ The probability block user from unauthorized process is greater than or equal 0.001.

$P_{\geq 0.0001} = [\diamond\ s = 14]$ The probability of hacking encryption key is greater than or equal 0.0001.

Continuous Time Markov Chain:

The requirements $R_{2,}\ R_{7,}\ R_{8}$

Below table demonstrate state and required time

Table 4.4 state rate times

	State rate	Value req\sec
1.	Login	20
2.	Main menu	20
3.	Select process	5
4.	Process success	20

The one security and failure rate requirement represented in figure 4.9

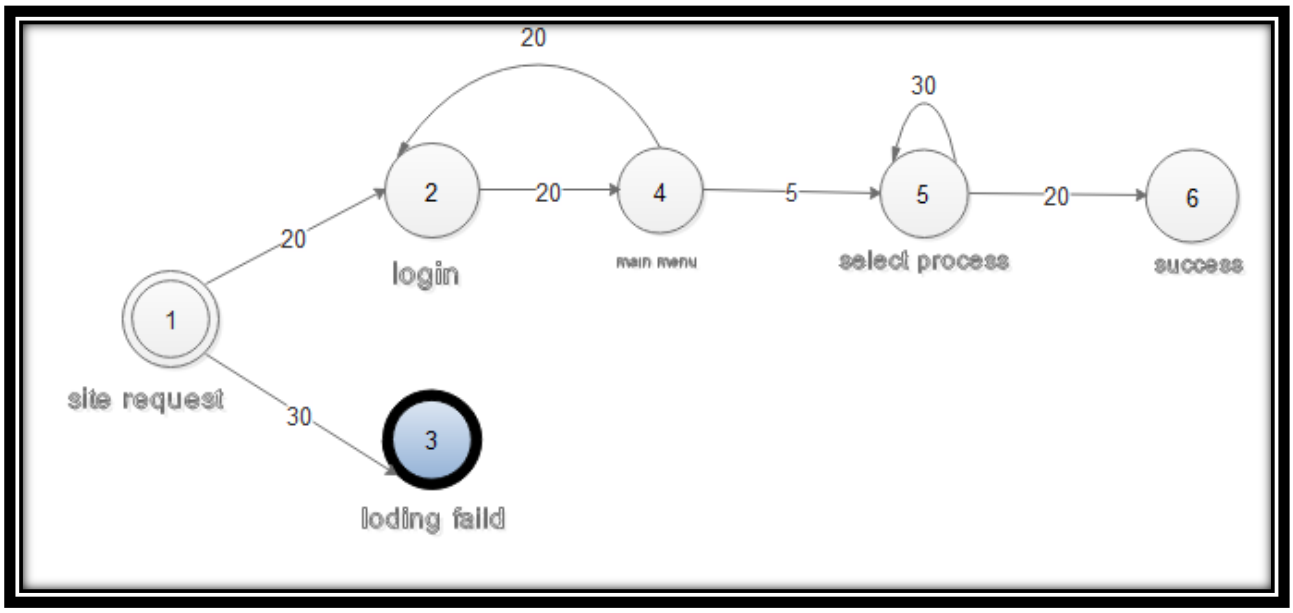


Figure 4.9 commercial registrars CTMC

The requirements can be translated to CSL as follow:

$P_{\leq 0.1} = [\diamond^{\geq 20}\ s = 2]$ The probability of system back to login state within 20 s is less than or equal 0.1

$P_{\leq 0.05} = [\diamond^{\geq 20}\ s = 3]$ The probability of loading site failed within 20 s is less than or equal 0.05

$P_{\leq 0.05} = [\diamond^{\leq 10}\ s = 5]$ The probability of system drop any of process and back to previous within 20s less than is less than or equal 0.05.

Use PRISM probabilistic model checkers to check the model properties.

1. PRISM module to check security and availability requirements:

```
dtmc

module comm_system

st:[1..14] init 1;

d:[5..14] init 5;

[]st=1->0.002:(st'=2)+0.499:(st'=3)+0.499:(st'=4);

[]st=2->0.25:(st'=3)+0.25:(st'=4)+0.5:(st'=5)&(d'=5);

[]st=3->1:( st'=6);

[]st=6->0.001:(st'=8)&(d'=8)+0.999:(st'=7);

[]st=7->0.8:(st'=11)&(d'=11)+0.001:(st'=10)&(d'=10)+0.100:(st'=9)+0.099:(st'=7);

[]st=9->0.001:(st'=13)&(d'=13)+0.999:(st'=12);

[]st=12->0.0001:(st'=14)&(d'=14)+0.9999:(st'=3);

endmodule
```

The results of experiments which gained from module according to properties when use PRISM explained in table 4.5:

Table 4.5 experiments results for security and availability

QoS requirement name	QoS requirements aspects	QoS probability value	Experiments result
Probability of hacker failed	Security Requirement	0.001	0.0010
Probability of loading site failed	Availability Requirement	0.001	5.617122544618364E-4
Probability user	Availability	0.0001	6.228085929049661E-4

verification failed	Requirement		
Probability of user lock	Security Requirement	0.8	0.49824687432397285
Probability of block user from unauthorized process	Security Requirement	0.001	6.228085929049662E-5
Probability of hacking encryption key	Security Requirement	0.0001	6.221857843120612E-6

The experiments result demonstrates the probabilities of verification are less than probabilities of QoS requirement, so the system satisfies user requirements.

In figure 4.10 the graph generated from run experiments, and another graphs generated from recurrent experiment will explain in appendix A.

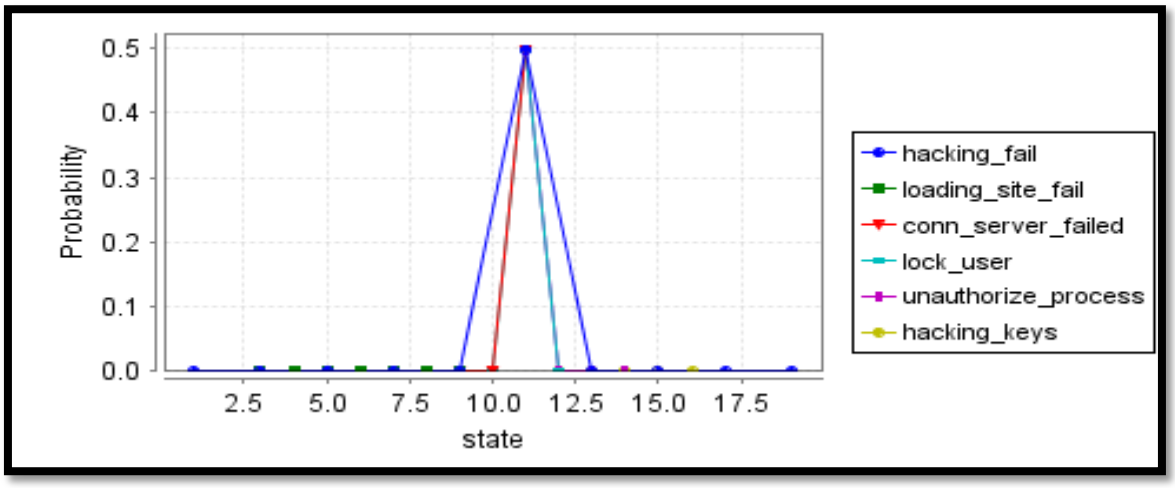


Figure 4.10 analyses security and availability

2. PRISM module to check security and failure rate requirements:

```
ctmc

module comm_ctmc_states

st:[1..6] init 1;

[]st=1->20:(st'=2)+30:(st'=3);

[]st=2->20:(st'=4);

[]st=4->5:(st'=5)+20:(st'=2);

[]st=5->20:(st'=6)+30:(st'=5);

endmodule
```

The results of experiments which gained from module according to properties when use PRISM explained in table 4.6:

QoS requirement name	QoS requirements aspects	QoS probability value	Experiments result
Probability of session time out	Security Requirements	<=0.01	0.4
Probability of loading site failed	Failure Requirements	0.05	0.6
Probability of drop process	Failure Requirements	0.05	0.4

Table 4.6 experiments results for security and Failure rate requirements

The result of experiment show that the probability of session time out is greater than the QoS requirement, this mean the system achieve the required QoS, the other probabilities is greater than QoS requirement, so the result probabilities show that the system not achieved all requirements.

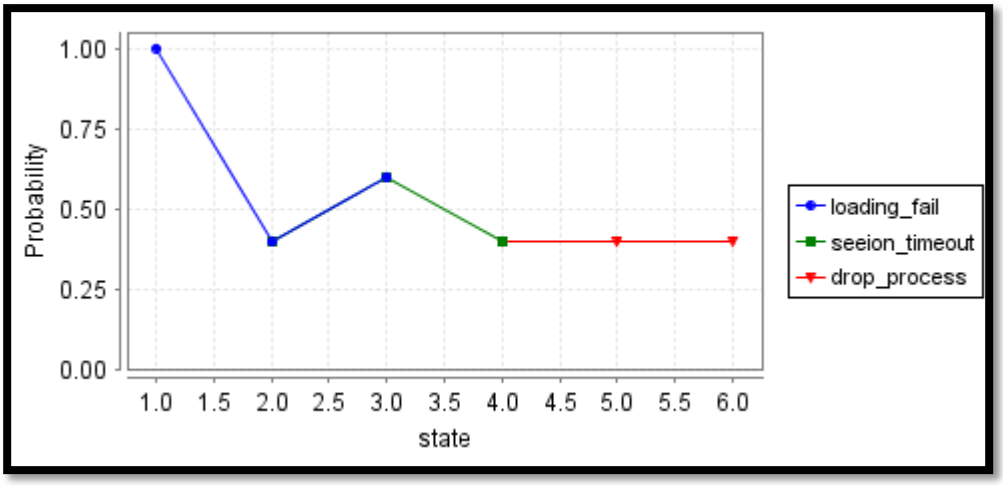


Figure 4.11 analyses of security and failure

4-4-Summary:

The descriptions of implementation and results mentioned in details to illustrate the stages of solution model which used to solve the research problem, and then the analyses of results to list of findings of implementation and evaluated the solution techniques and accuracy comparisons with many other techniques can use to solve the same problems.

CHAPTER IV

Conclusion

5-1-Conclusion:

The importance of quality was became clear in all services area and systems running on these area, loss of quality lead to disasters, some fields measure the quality according standard metrics specify by experts at these fields, while athor fields measure the quality according to multi factors, most popular factor is user satisfaction level, at these fields the systems must present services as required by user, so the services always was intended according to user requirements, this means user satisfaction level is the most important factor to measure QoS .

This study was intended to provide solution for the validity of QoS after requirements specification, to achieve quality verification we used model checking technique and PRISM tool as checker software, by using these technique built verification model for validity of QoS requirements. The QoS which verified was specified from system requirements; these requirements were consider is QoS requirements, The probabilistic model checking is useful in debugging system to discover the systems bugs, so when we use in verification, in addition to validity of QoS requirements, it support to discovering which requirements violated specifically, this is useful when the system redeveloped because facilitate the task of developing , the develop will achieved for requirements which violated.

The output of verification is more accurate because it depend on system states and probability of navigate between states; the states of system represent the QoS requirements, so the method used to solve this problem is more flexible because all QoS features can represented as requirements and then validated, some published studies concentrate on certain QoS aspect rather than other, the aspects specify according to system need to measure quality for. The validation for QoS achieved for the system.

The validation outputs explain that the QoS not achieved for all system requirements, there some requirements not achieved as are should, and that mean the system must redevelop to meet all QoS requirements.

From previous mentioned the contributions of research as follow:

1. Build verification model for QoS using probabilistic model checking.
2. Validity of QoS requirements for any application using built model.
3. Discover which QoS requirements not achieved as required
4. Build standard document to develop system in future.

5-2-Recomondations:

To avoid recurrent system development after each QoS validity, is best if two processes achieved in parallel, means during system design stage consider QoS Requirements; when need to design system for achieve service, firstly must study the environment which will operate in, and customers of service, and then study standard of QoS and affected by these two factors . By using this method can save time and produce accurate systems that implicate QoS guarantee, because considering customers of service lead to knowledge of user experience and considering it in future development of system, and study system environment support for design scalable system.

Reference:

[1] Dr. habil. Konrad Froitzheim, and Prof. David Hutchison, "QoS based Real-Time Audio Streaming in the Internet".

[2] Groenda,and Henning, "Certifying Software Component Performance Specifications".

[3] Daniel G. Canton-Puerto, Francisco Moo-Mena, and Víctor Uc-Cetina , "QoS-Based Web Services Selection Using a Hidden Markov Model" , Journal of Computers, December 27, 2015, pp.48-56.

[4] P. Calduwel Newton, and L. Arockiam,"A Quality of Service Performance Evaluation Strategy for Delay Classes in General Packet Radio Service", International Journal of Advanced Science and Technology , January, 2013, pp. 91- 98.

[5] J. Schiller, "Mobile Communications", Second Edition, Pearson Education,2003.

[6] Elarbi Badidi , "A Broker-Based Framework For Integrated SLA-Aware SAAS Provisioning", International Journal on Cloud Computing: Services and Architecture (IJCCSA) Vol. 6, No. 2, April 2016.

[7] Dimitrios Miras, "Network QoS Needs of Advanced Internet Applications", page 2.

[8] Linlin Wu, Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya, "SLA-Based Resource Provisioning for Hosted Software-as-a-Service Applications in Cloud Computing Environments", Ieee Transaction on Services Computing, VOL. 7, NO. 3, JULY-SEPTEMBER 2014, pp. 465-485.

[9] Torsten Braun and Thomas Staub, "End-to-End Quality of Service Over Heterogeneous Networks", 2008.

[10] José Darío Luis Delgado and Jesús Máximo Ramírez Santiago, "Key Performance Indicators for QoS Assessment in TETRA Network", International Journal of Mobile Network Communications & Telematics (IJMNCT) Vol. 3, No.6, December 2013.

[11] Fei Peng, Xuewen Zeng, Haojiang Deng and Lei Liu, "The QoS Prediction of Web Service with Location Information Ensemble", Journal of Software , VOL. 9, NO. 5, May 2014, pp. 1210-1216.

[12] Amid Khatibi Bardsiri and Seyyed Mohsen Hashemi, "QoS Metrics for Cloud Computing Services Evaluation", I.J. Intelligent Systems and Applications, 2014, 12, pp.27-33

[13] Chen Wu, Weiwei Qiu, Zibin Zheng, Xinyu Wang and Xiaohu Yang, "QoS Prediction of Web Services based on Two-Phase K-Means Clustering", IEEE International Conference on Web Services,2015, pp. 161-168.

[14] Gerasimou, Simos, Tamburrelli, Giordano and Calinescu, Radu, "Search-Based Synthesis of Probabilistic Models for Quality-of-Service Software Engineering", 30th IEEE/ACM international Conference on Automated Software Engineering (ASE 2015).

[15] Lata Nautiyal and Preeti, " Evaluating and Certifying Component-Based Software Using Weighted Assignment Technique", International Journal of Hybrid Information Technology Vol.9, No.1, 2016, pp. 241-252.
[16] Md. I. Hussain, and N. Ahmed, "A Performance Analysis of E-Learning over WiFi-based Long Distance Networks", Journal of Wireless Networking and Communications 2016, 6(4), pp.85-93.
[17] Pengcheng Zhang, Qing Han, Wenrui Li, Hareton Leung,and Wei Song, "A Novel QoS Prediction Approach for Cloud Service Based on Bayesian Networks Model", IEEE International Conference on Mobile Services,2016, pp.111-118.
[18] Yan Guo, Shangguang Wang, Kok-seng Wong and Myung Ho Kim, "Skyline Service Selection Approach based on QoS Prediction", International Journal of Web and Grid Services, Vol. , No. , 2016.
[19] Hua Ma, Haibin Zhu, Zhigang Hu, Wensheng Tang,and Pingping Dong, "Multi-valued collaborative QoS prediction for cloud service via time series analysis", Future Generation Computer Systems 68,2017, pp.275–288.
[20] R. Sarala , P. Manisha, Mukku Vineesha, and G. Indumathy, " A Consumption History and QoS based Web Service Ranking Technique", International Conference on Emerging Innovation in Engineering and Technology,March 2017,pp 15-20.
[21] Rafid Mustafiz, Abu Sayem Mohammad Delowar Hossain, Nazrul Islam, and Mohammad Motiur Rahman, "Analysis of QoS in Software Defined Wireless Network with Spanning Tree Protocol", I. J. Computer Network and Information Security, june 2017, pp. 61-68.
[22] Ababakr Ibrahim Rasul, Diyar Salah Fadhil and Younus Ameen Muhammed, "The Performance Evaluation of Voip for a Mobile User", Society for Science and Education United Kindoum, June 2017, pp 21 -24.
[23] Jian-Long XU, and Chang-Sheng ZHU, "Personalized and Accurate QoS Prediction Approach Based on Online Learning Matrix Factorization for Web Services", ITM Web of Conferences, 2017.
[24]Jalel Eddine HAJLAOUI, Mohamed Nazih OMRI, and Djamal BENSLIMANE, "A QoS-aware approach for discovering and selecting configurable IaaS Cloud services",researchgate.net
[25] Shangguang Wang, Yali Zhao, Lin Huang, Jinliang Xu1,and Ching-Hsien Hsu, "QoS Prediction for Service Recommendations in Mobile Edge Computing".
[26]Jieming Zhu, Pinjia He, Zibin Zheng, and Michael R. Lyu, "Online QoS Prediction for Runtime Service Adaptation via Adaptive Matrix Factorization", Ieee Transactions on Parallel and Distributed Systems, 2017.
[27] Gary White, Andrei Palade, Christian Cabrera,and Siobh´an Clarke, "Quantitative Evaluation of QoS Prediction in IoT", IEEE/IFIP International Conference on Dependable Systems and Networks Workshops, 2017 .

[28] Fabrizio Messina ,Giuseppe Pappalardo, Antonello Comi, Lidia Fotia, Domenico Rosaci and Giuseppe M.L. Sarné**, "**Combining reputation and QoS measures to improve cloud service composition", Int. J. Grid and Utility Computing, 2017, pp. 142-152.
[29]Chandrashekar Jatoth , G.R. Gangadharan, Ugo Fiore, and Rajkumar Buyya, "QoS-aware Big service composition using MapReduce based evolutionary algorithm with guided mutation"
, Future Generation Computer Systems July,2017.
[30] Ireneusz Jozwiak, Michal Kedziora and Aleksander Marianski , "Service selection method with multiple probabilistic QoS attributes using probabilistic AHP", IJCSNS International Journal of Computer Science and Network Security,2018, PP 33-38
[31]Hong Ji, Xifan Yao , and Yong Chen**,"** Correlation-aware QoS modeling and manufacturing cloud service composition", Springer Science Business Media New York 2015.
[32] Kirit J. Modi, and Sanjay Garg, "Dynamic Web Services Composition using Optimization Approach", IJCSC, Sep 2015 pp. 285-293.
[33]Pablo Rodriguez-Mier, Manuel Mucientes, and Manuel Lama, "Hybrid Optimization Algorithm for Large-Scale QoS-Aware Service Composition", IEEE Transaction on Services Computing ,2015, PP. 1-17.
[34]Amina BEKKOUCHE, Sidi Mohammed BENSLIMANE, Marianne HUCHARD, Chouki TIBERMACINE, Fethallah, HADJILA, and Mohammed MERZOUG, "QoS-Aware Optimal and Automated Semantic Web Service Composition With User's Constraints".
[35] Yan Guo, Shangguang Wang, Kok-seng Wong and Myung Ho Kim, "Skyline Service Selection Approach based on QoS Prediction", International Journal of Web and Grid Services, Vol. , No. , 2016.
[36] Diego Perez-Palacin, Raffaela Mirandola, and Jos´e Merseguer, "Accurate Modeling and Efficient QoS Analysis of Scalable Adaptive Systems under Bursty Workload", Journal of Systems and Software May 2, 2017.
[37] Jalel Eddine HAJLAOUI, Mohamed Nazih OMRI, and Djamal BENSLIMANE, "A QoS-aware approach for discovering and selecting configurable IaaS Cloud services",researchgate.net
[38]Samia Sadouki Chibani and Abdelkamel Tari, "Elephant Herding Optimization for Service
Selection in QoS-Aware Web Service Composition", International Journal of Computer and Information Engineering, 2017, pp. 1116-11120.
[39]Elsoon Neshati and Ali Asghar Pourhaji Kazem, "QoS-based Cloud Manufacturing Service Composition using Ant Colony Optimization Algorithm", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 9, No. 1, 2018, pp.437-440.
[40] Ashkan Yousefpour, Ashish Patil, Genya Ishigaki, Jason P. Jue, Inwoong Kimy, Xi Wangy, Hakki C. Cankaya, Qiong Zhangy and Weisheng Xie, "QoS-aware Dynamic Fog Service Provisioning", arXiv:1802.00800v1 [cs.NI] 2 Feb 2018.

[41] Savas Konur, "Real-time and Probabilistic Temporal Logics": An Overview November 2008.
[42] Marie Duflot, Marta Kwiatkowska, Gethin Norman, David Parker, Sylvain Peyronnet, Claudine Picaronny, and Jeremy Sproston, "Practical Applications of Probabilistic Model Checking to Communication Protocols"
[43] Frederic Havet, Introduction to the probabilistic method, 2008.
[44] Chanyeol Yoo, Robert Fitch and Salah Sukkarieh, "Probabilistic Temporal Logic for Motion Planning with Resource Threshold Constraints".
[45] Hakan Lorens Samir Younes, Verification and Planning for Stochastic Processes with Asynchronous Events, 2005, pp.46.
[46] Christel Baier, Luca de Alfaro, Vojtˇech Forejt, and Marta Kwiatkowska, "Probabilistic Model Checking".
[47] Daniel Wagner, "Finite-State Abstractions for Probabilistic Computation Tree Logic", Imperial College London, Department of Computing, 2011.
[48] Marta Kwiatkowska, and David Parker, "Advances in Probabilistic Model Checking".
[49] Koushik Sen, Mahesh Viswanathan, and Gul Agha, "On Statistical Model Checking of Stochastic Systems".
[50] Christel Baiera, Joost-Pieter Katoenb, Holger Hermannsb, and Boudewijn Haverkortc, "Simulation for Continuous-Time Markov Chains".
[51] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf, "Three-Valued Abstraction for Continuous-Time Markov Chains.
[52] Sigman Karl, and Ronald W. Wolff. "A Review of Regenerative Processes." *SIAM Review*, vol. 35, no. 2, 1993, pp. 269–288.
[53] Marta Kwiatkowska ,Gethin Norman and David Parker , "Quantitative analysis with the probabilistic model checker PRISM", QAPL 2005.
[54] Takashi Tomita, Shigeki Hagihara and Naoki Yonezaki, A Probabilistic Temporal Logic with Frequency Operators and Its Model Checking.
[55] Marta Kwiatkowska, Gethin Norman, and David Parker, "PRISM: Probabilistic Symbolic Model Checker".
[56]Alexey Ilyushkin, Ahmed Ali-Eldin, Nikolas Herbst,Alessandro V. apadopoulos, Bogdan Ghit,, Dick Epema, and Alexandru Iosup, **"**An Experimental Performance Evaluation of Autoscaling Policies for Complex Workflows", 2017, ACM.
[57]Diego Perez-Palacin, Raffaela Mirandola, and Jos´e Merseguer, "Accurate Modeling and Efficient QoS Analysis of Scalable Adaptive Systems under Bursty Workload", Journal of Systems and Software May 2, 2017.
[58] Christos Chrysoulas, and Maria Fasli, "Towards an adaptive SOA-based QoS & Demand-Response Provisioning Architecture for the Smart Grid", Journal of Communications Software and Systems, VOL. 13, NO. 2, june 2017,pp.77-86.
[59]Wen-long ZHU, Bei-bei YIN,Si-qian GONG and Kai-Yuan CAI, **"**An Advanced QoS-Based Web Service Selection Approach", International Conference on Computer Science and Technology (CST 2017) pp 906 – 917.

[60] MA Hua, HU Zhigang1 and CAI Meiling, “Trustworthy Service Selection Integrating Cloud Model and Possibility Degree Ranking of Interval Numbers”, Chinese Journal of Electronics (2017), pp 1177-1183.
[61] Yuyu Yin, Wenting Xu, Yueshen Xu, He Li, and Lifeng Yu, **“**Collaborative QoS Prediction for Mobile Service with Data Filtering and SlopeOne Model”, Mobile Information Systems, 2017.
[62]Soumi Chattopadhyay and Ansuman Banerjee,” QoS constrained Large Scale Web Service Composition using Abstraction Refinement”, arXiv:1608.08799v2 [cs.SE] 15 Feb 2017.
[63] Sepideh Sheivandi, and Sima Emadi, “Automatic Service Composition Based on Graph Coloring”, Journal of Advances in Computer Research Quarterly 2017 pp. 91-103.

Appendix A:

Here will include more than one graph and result for experiments result for each representation types of systems.

1. Case study no1:

The result for all experiments for DTMC for the system include at the feature of PRISM module execution

```
"P=? [ F ds=x&fs=x ]:"
x, Result
1, 0.0
2, 0.0
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
```

```
30, 0.0
31, 0.0
32, 0.0
33, 0.0
34, 0.0
35, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
2, 0.0
4, 0.018000000000000002
6, 0.01764
8, 0.0
10, 0.0
12, 0.0
14, 0.0
16, 0.0
18, 0.0
20, 0.0
22, 0.0
24, 0.0
26, 0.0
28, 0.0
30, 0.0
32, 0.0
34, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
4, 0.018000000000000002
6, 0.01764
8, 0.0
10, 0.0
12, 0.0
14, 0.0
16, 0.0
18, 0.0
20, 0.0
22, 0.0
24, 0.0
26, 0.0
```

```
28, 0.0
30, 0.0
32, 0.0
34, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
2, 0.0
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
30, 0.0
31, 0.0
32, 0.0
33, 0.0
34, 0.0
35, 0.0
```

"P=? [F ds=x&fs=x]:"
x, Result
2, 0.0
4, 0.018000000000000002
6, 0.01764
8, 0.0
10, 0.0
12, 0.0
14, 0.0
16, 0.0
18, 0.0
20, 0.0
22, 0.0
24, 0.0
26, 0.0
28, 0.0
30, 0.0
32, 0.0
34, 0.0

"P=? [F ds=x&fs=x]:"
x, Result
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0

```
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
30, 0.0
31, 0.0
32, 0.0
33, 0.0
34, 0.0
35, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
```

```
30, 0.0
31, 0.0
32, 0.0
33, 0.0
34, 0.0
35, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
1, 0.0
2, 0.0
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
30, 0.0
31, 0.0
32, 0.0
```

```
33, 0.0
34, 0.0
35, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
3, 0.1
4, 0.0180000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
30, 0.0
31, 0.0
32, 0.0
33, 0.0
34, 0.0
35, 0.0

"P=? [ F ds=x&fs=x ]:"
```

```
x, Result
1, 0.0
2, 0.0
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
30, 0.0
31, 0.0
32, 0.0
33, 0.0
34, 0.0
35, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
2, 0.0
4, 0.018000000000000002
```

```
6, 0.01764
8, 0.0
10, 0.0
12, 0.0
14, 0.0
16, 0.0
18, 0.0
20, 0.0
22, 0.0
24, 0.0
26, 0.0
28, 0.0
30, 0.0
32, 0.0
34, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
4, 0.0180000000000000002
6, 0.01764
8, 0.0
10, 0.0
12, 0.0
14, 0.0
16, 0.0
18, 0.0
20, 0.0
22, 0.0
24, 0.0
26, 0.0
28, 0.0
30, 0.0
32, 0.0
34, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
0, 0.0
1, 0.0
2, 0.0
3, 0.1
```

```
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0
21, 0.0
22, 0.0
23, 0.0
24, 0.0
25, 0.0
26, 0.0
27, 0.0
28, 0.0
29, 0.0
30, 0.0
31, 0.0
32, 0.0
33, 0.0
34, 0.0
35, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
2, 0.0
4, 0.018000000000000002
6, 0.01764

"P=? [ F ds=x&fs=x ]:"
x, Result
```

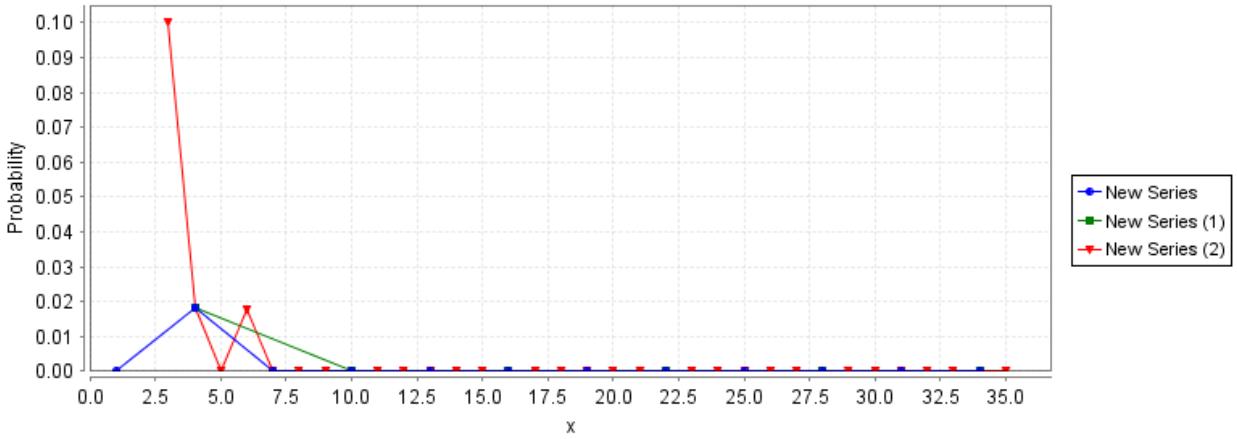
```
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
4, 0.018000000000000002
6, 0.01764
8, 0.0
10, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
0, 0.0
1, 0.0
2, 0.0
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
```

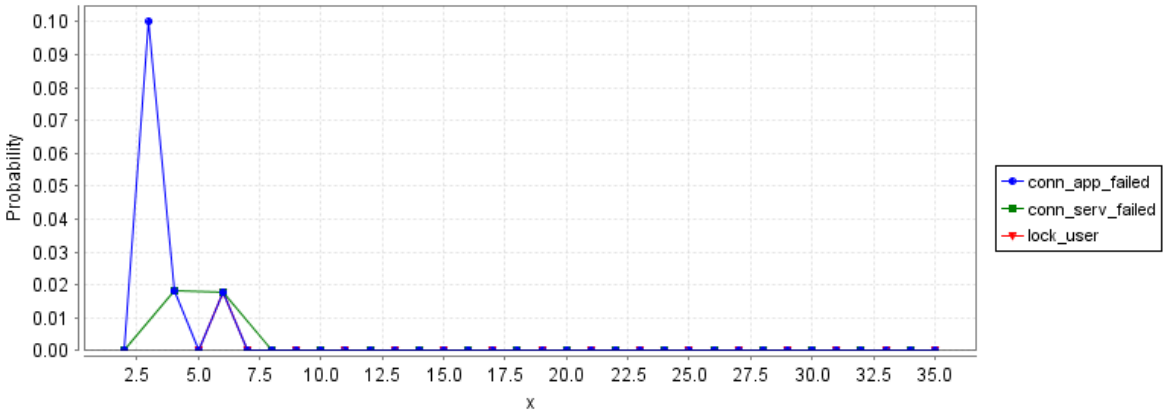
```
2, 0.0
4, 0.018000000000000002
6, 0.01764
8, 0.0
10, 0.0
12, 0.0
14, 0.0
16, 0.0
18, 0.0
20, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
3, 0.1
5, 0.0
7, 0.0
9, 0.0
11, 0.0
13, 0.0
15, 0.0
17, 0.0
19, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
0, 0.0
1, 0.0
2, 0.0
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
```

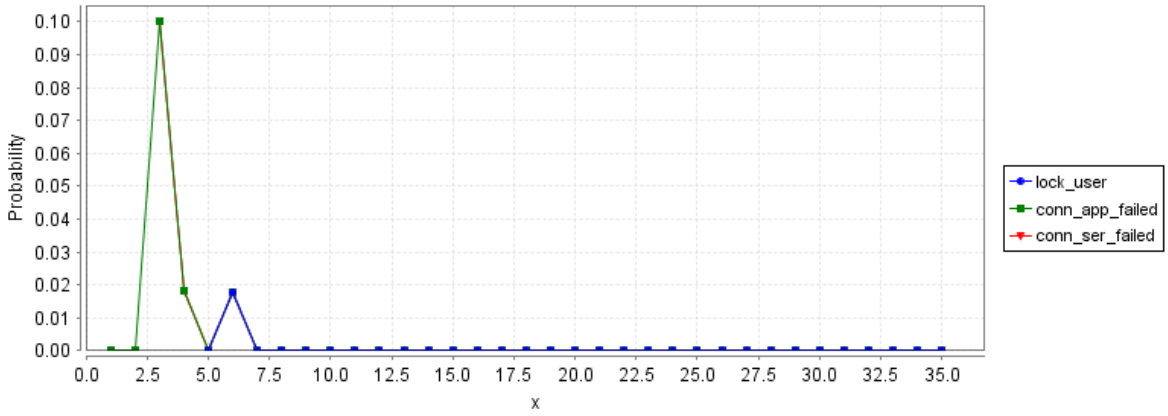
```
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
2, 0.0
4, 0.018000000000000002
6, 0.01764

"P=? [ F ds=x&fs=x ]:"
x, Result
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
8, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
1, 0.0
3, 0.1
5, 0.0
7, 0.0
9, 0.0
11, 0.0
13, 0.0
15, 0.0
17, 0.0
19, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
3, 0.1
4, 0.018000000000000002
5, 0.0
6, 0.01764
7, 0.0
```

```
8, 0.0
9, 0.0
10, 0.0
11, 0.0
12, 0.0
13, 0.0
14, 0.0
15, 0.0
16, 0.0
17, 0.0
18, 0.0
19, 0.0
20, 0.0

"P=? [ F ds=x&fs=x ]:"
x, Result
0, 0.0
2, 0.0
4, 0.018000000000000002
6, 0.01764
```

The graphs generated from these result below:







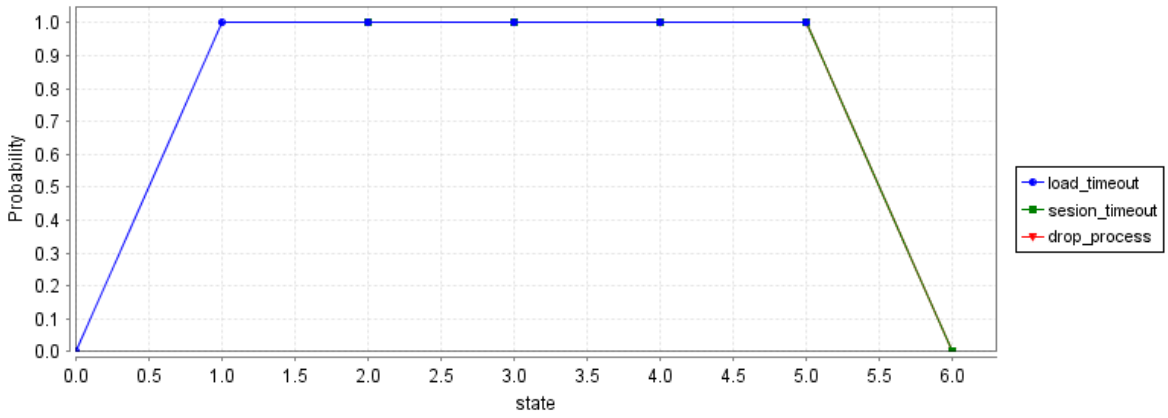
The results for CTMC module:

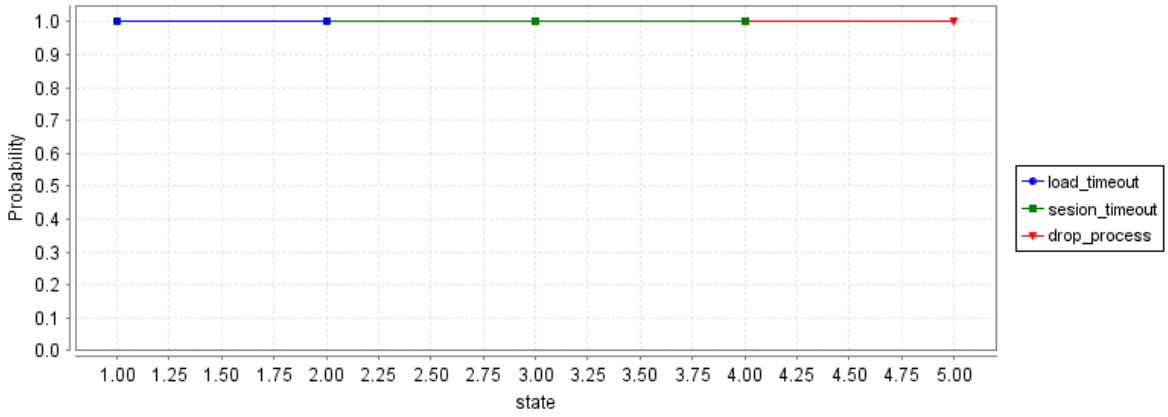
```
P=? [ F st=state ]:
state      Result
0     0.0
1     1.0
2     1.0
3     1.0
4     1.0
5     1.0
6     0.0
7     0.0
8     0.0
9     0.0
10    0.0
11    0.0
12    0.0
13    0.0
14    0.0
```

```
15   0.0
16   0.0
17   0.0
18   0.0
19   0.0
20   0.0

P=? [ F st=state ]:
state     Result
2    1.0
4    1.0
6    0.0
8    0.0
10   0.0
12   0.0
14   0.0
16   0.0
18   0.0
20   0.0

P=? [ F st=state ]:
state     Result
3    1.0
4    1.0
5    1.0
6    0.0
7    0.0
8    0.0
9    0.0
10   0.0
11   0.0
12   0.0
13   0.0
14   0.0
15   0.0
16   0.0
17   0.0
18   0.0
19   0.0
20   0.0
```

```
P=? [ F st=state ]:
state     Result
1    1.0
2    1.0

P=? [ F st=state ]:
state     Result
1    1.0
2    1.0
3    1.0
4    1.0

P=? [ F st=state ]:
state     Result
4    1.0
5    1.0

P=? [ F st=state ]:
state     Result
0    0.0
1    1.0
2    1.0
3    1.0
4    1.0
5    1.0

P=? [ F st=state ]:
state     Result
2    1.0
3    1.0
4    1.0
5    1.0
6    0.0

P=? [ F st=state ]:
state     Result
4    1.0
5    1.0
6    0.0
```

The graphs generated from these results are below:





1. Case study no2:

   The result as gain from prism modules execution and graphs generated from results:

   DTMC module result:

```
P=? [ F st=state&d=state ]:
state      Result
1     0.0
2     0.0
3     0.0
4     0.0
5     0.0010
6     0.0
7     0.0
8     5.617122544618364E-4
9     0.0
```

```
10   6.228085929049661E-4
11   0.49824687432397285
12   0.0
13   6.228085929049662E-5
14   6.221857843120612E-6
15   0.0
16   0.0
17   0.0
18   0.0
19   0.0
20   0.0
21   0.0
22   0.0
23   0.0
24   0.0
25   0.0
26   0.0
27   0.0
28   0.0
29   0.0
30   0.0
31   0.0
32   0.0
33   0.0
34   0.0
35   0.0

P=? [ F st=state&d=state ]:
state     Result
6    0.0
7    0.0
8    5.617122544618364E-4
9    0.0
10   6.228085929049661E-4

P=? [ F st=state&d=state ]:
state     Result
8    5.617122544618364E-4
9    0.0
10   6.228085929049661E-4
11   0.49824687432397285
```

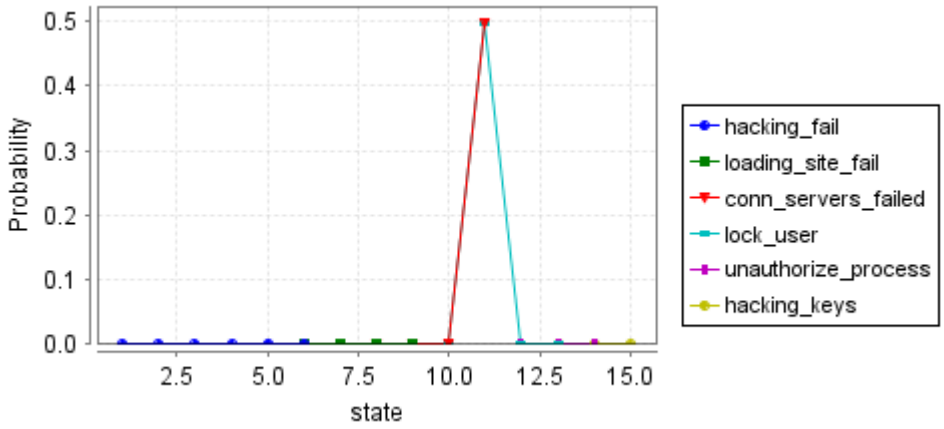
```
12   0.0
13   6.228085929049662E-5
14   6.221857843120612E-6

P=? [ F st=state&d=state ]:
state     Result
9    0.0
10   6.228085929049661E-4
11   0.49824687432397285
12   0.0
13   6.228085929049662E-5
14   6.221857843120612E-6

P=? [ F st=state&d=state ]:
state     Result
11   0.49824687432397285
12   0.0
13   6.228085929049662E-5
14   6.221857843120612E-6

P=? [ F st=state&d=state ]:
state     Result
12   0.0
13   6.228085929049662E-5
14   6.221857843120612E-6

P=? [ F st=state&d=state ]:
state     Result
1    0.0
2    0.0
3    0.0
4    0.0
5    0.0010
6    0.0

P=? [ F st=state&d=state ]:
state     Result
6    0.0
7    0.0
8    5.617122544618364E-4
9    0.0
```

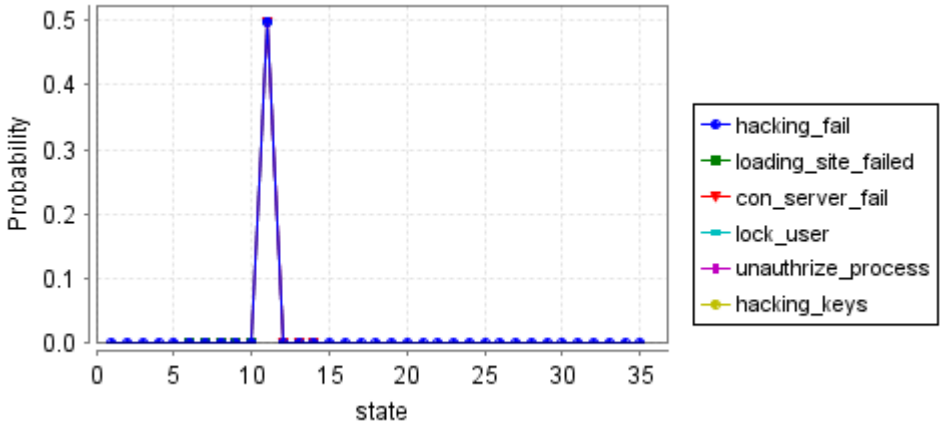
```
P=? [ F st=state&d=state ]:
state     Result
8    5.617122544618364E-4
9    0.0
10   6.228085929049661E-4
11   0.49824687432397285

P=? [ F st=state&d=state ]:
state     Result
9    0.0
10   6.228085929049661E-4
11   0.49824687432397285
12   0.0
13   6.228085929049662E-5

P=? [ F st=state&d=state ]:
state     Result
12   0.0
13   6.228085929049662E-5
14   6.221857843120612E-6

P=? [ F st=state&d=state ]:
state     Result
13   6.228085929049662E-5
14   6.221857843120612E-6
15   0.0

P=? [ F st=state&d=state ]:
state     Result
1    0.0
3    0.0
5    0.0010
7    0.0
9    0.0
11   0.49824687432397285
13   6.228085929049662E-5
15   0.0
17   0.0
19   0.0
```

```
P=? [ F st=state&d=state ]:
state     Result
1    0.0
3    0.0
5    0.0010
7    0.0
9    0.0
11   0.49824687432397285
13   6.228085929049662E-5
15   0.0
17   0.0
19   0.0

P=? [ F st=state&d=state ]:
state     Result
3    0.0
4    0.0
5    0.0010
6    0.0
7    0.0
8    5.617122544618364E-4
9    0.0

P=? [ F st=state&d=state ]:
state     Result
9    0.0
10   6.228085929049661E-4
11   0.49824687432397285

P=? [ F st=state&d=state ]:
state     Result
10   6.228085929049661E-4
11   0.49824687432397285
12   0.0

P=? [ F st=state&d=state ]:
state     Result
11   0.49824687432397285
12   0.0
13   6.228085929049662E-5
14   6.221857843120612E-6
```

```
P=? [ F st=state&d=state ]:
state     Result
12    0.0
13    6.228085929049662E-5
14    6.221857843120612E-6
15    0.0
16    0.0
```

The graphs generated are:





The result from CTMC module is:

```
P=? [ F st=state ]:
state     Result
     1          1
```

2	0.4
3	0.6
4	0.4
5	0.4
6	0.4
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
29	0
30	0

P=? [F st=state]:

state	Result
2	0.4
3	0.6
4	0.4

P=? [F st=state]:

state	Result

5	0.4
6	0.4

P=? [F st=state]:

state	Result
1	1
2	0.4
3	0.6

P=? [F st=state]:

state	Result
2	0.4
3	0.6
4	0.4

P=? [F st=state]:

state	Result
4	0.4
5	0.4
6	0.4

P=? [F st=state]:

state	Result
0	0
1	1
2	0.4
3	0.6
4	0.4
5	0.4
6	0.4

P=? [F st=state]:

state	Result
2	0.4
3	0.6
4	0.4
5	0.4
6	0.4

P=? [F st=state]:

state	Result
1	1
2	0.4
3	0.6

P=? [F st=state]:

state	Result
1	1
2	0.4

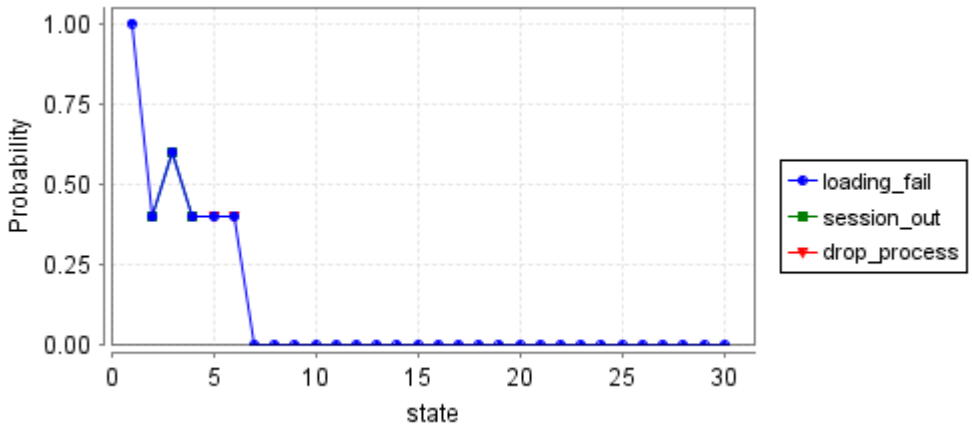
P=? [F st=state]:

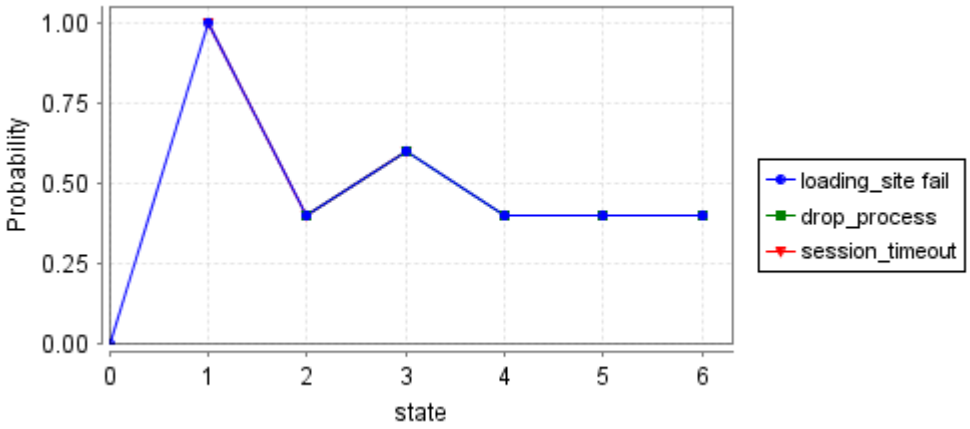
state	Result
1	1
2	0.4
3	0.6

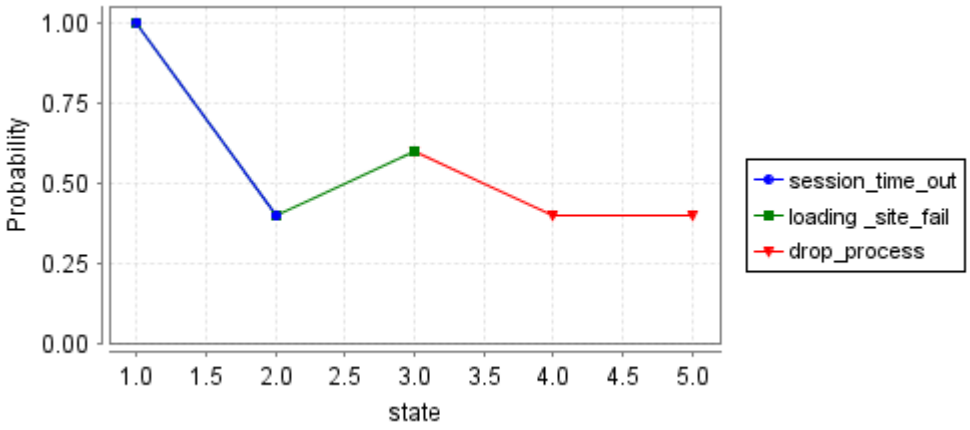
P=? [F st=state]:

state	Result
3	0.6
4	0.4
5	0.4

The graphs generated are:




1.00
0.75
0.50
0.25
0.00
Probability
0
1
2
3
4
5
6
state
loading_site fail
drop_process
session_timeout


1.00
0.75
0.50
0.25
0.00
Probability
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
state
session_time_out
loading _site_fail
drop_process