Performance Prediction for Web Services Based on Dynamic Workload: A Simulation Approach

Ch. Ram Mohan Reddy*, D. Evangelin Geetha*, R. V. Raghavendra Rao*, and K. Sailaja Kumar*
B. M. S. College of Engineering, Bangalore-19, INDIA
Ramaiah Institute of Technology, Bangalore - 54, INDIA

Abstract

The increase in the use of web services has rendered its workload to be highly uncertain. The workload and the execution environment of web services affect their performance since they are non-uniform and unpredictable. Hence, estimation techniques are required to predict the performance of web services considering their dynamic workload. This paper presents a methodology to estimate the response time of web services over a given time horizon by providing a mathematical model. The deployment environment is analyzed by considering the configuration of the resources and the workload that fluctuates over a period of time. A case study on Travel care application is presented to illustrate the methodology. The tool SMTQA (Simulation of Multi-Tier Queuing Applications) is used to carry out sensitivity analysis on the configuration of resources so that the behavior of the resources can be analyzed. This analysis helps to improve the performance of web services by identifying the bottleneck resources. Moreover, it provides a possibility to determine the performance objectives.

Key Words: Web services, software performance engineering, performance goal, dynamic workload, mathematical model, sensitivity analysis, SMTQA.

1 Introduction

One of the most significant aspects of Quality of Service (QoS) is performance. It is critical to evaluate the aspects that lead to client satisfaction in the efficient delivery of various services. Certainly, improving these factors is a challenge in web services since the workload from the client-side is uncertain and ever-changing. The performance of web services can be analyzed in several ways. It is necessary to analyze the workload to meet users' performance expectations.

Composite services are a typical aggregation of the complex process. In the context of web services, a composite web service is considered as a single logical unit [10]. The performance of the composite web services encompasses the effective resource usage and sharing of the workload in a service-oriented environment. It is difficult to know the workload of the composite web services. Any online service offered on the internet can be evaluated by looking at four key factors: number, service quality, complexity, and function diversity [11].

Two important performance dimensions for web applications are responsiveness and scalability [2, 18]. In the current scenario, the web users are too busy to wait for a slow responsive system; hence, responsiveness is significant. Scalability is important to maintain responsiveness, as more and more users converge on a site. There is also significant capacity planning concerns for web applications, such as the selection of the number of processing nodes, the number of processors for each node, the speed of the processors, and so on. Other significant performance tuning issues also have to be addressed for responsive web applications. One of the critical issues for the performance of web services is the issue of balancing the workload of computational tasks among the different nodes comprising the system [15].

Early in the software development process, Software Performance Engineering (SPE) provides numerous approaches for analyzing the performance of software systems. It is difficult to obtain reliable early estimations because complete information about the future system is lacking at this time. However, to establish the feasibility of a software system in terms of performance analysis, early estimates are essential [18-19].

A methodology to analyze the behavior of the hardware resources based on the dynamic workload is proposed in this paper. The methodology provides a basis for calculating the response time over a given time horizon based on the demand of the activities of the web services. Moreover, the methodology helps to define the performance goal as well.

2 Related Work

As discussed in the previous section, the non-uniform workload of web services has a strong impact on system performance. Some literature available in this context is reviewed.

The Customer Behavior Model Graph (CBMG) is a graph that describes patterns of customer behavior in e-commerce site workloads [14, 20]. The impact of a more realistic dynamic workload on online performance measures is investigated in [16]. The analysis is done by carrying out an experimental study on an e-commerce scenario with a dynamic workload. The obtained results are compared with the traditional workloads. In multi-tier web service systems, a soft resource allocation approach is proposed to handle dynamic workloads in real-time [25]. The authors have formulated the whole system by queueing the network model. To cope with dynamic workloads and to meet performance demands, an optimization approach based on sliding windows

* Emails: prof.crmr@gmail.com, degeetha@msrit.edu, rvraghavendrarao76@gmail.com, sailajakumar.k@gmail.com.

is presented. Experimental measurements are used to validate the effectiveness of the optimization method, as well as the model parameters and performance indicators. The author of [23] proposes a method for both economical and robust provisioning of resources for N-tier web applications. The authors tested the method using three different workload models in the RUBiS web application benchmark: open, closed, and semi-open. The authors claimed that their approach was flexible enough to accommodate a wide variety of workloads with varying resource demands without requiring reconfiguration.

Web service compositions are emerging to support industries by becoming more dynamic and delivering customized services to users. Dynamic web service composition approaches as the foundation of problems like transaction support, compositional correctness, etc. are compared in [9]. In [3] the performance of the clinical decision support system is shown based on a web service. The layered queueing network model is followed to design the software architecture of the system, and the systems performance goals are obtained by solving the model analytically. E-commerce applications are evaluated using the queuing network model in [8]. Through the use of techniques like Layered Queuing Network (LQN) models and SPE, performance parameters such as response time, utilization, and throughput have been verified with actual measurements. In [21] an architecture behavior designed for varying demands is discussed that are placed on the server. The paper discusses the experimental studies on an infrastructure behavior that is devised for variant loads placed on the server.

Web services performance evaluation can be done at two levels. One is at the client-side which is a direct interface with the user and the other at the server-side. The performance has to be very good at the client-side, and various factors that contribute to the effective performance of web services have to be considered as in [12]. The performance of a Web service has to be estimated properly when we integrate different web services to make a new one. Different web services can be combined based on simplicity, interoperability, flexibility, and reuse of services. Different toolkits are used to compare the performance of Integrated Web Services. The authors describe a new concept of creating web services, based on existing services to enhance the performance of a web service, also called a Hybrid approach [4].

To handle the growing client and varying workload, a session-based workload and reliability analysis is presented in [22]. The authors introduced intra-session and inter-session metrics. Describing workload and analyzing characteristics of web errors by estimating request-based and session-based web server reliability to indicate user perception is discussed. Performance analysis using a measurement-based performance analysis is explained with an example of an e-commerce application that uses a web service component in [5]. According to the setup established, higher workload intensities increase the response time of the web services in comparison with any other component. A methodology for selecting an appropriate execution environment based on dynamic workloads over a given time horizon is discussed in [7]. The authors propose a mathematical model for calculating the performance which facilitates decision-making for a given distributed application by calculating its performance. It also simulates the model proposed for the same.

In the literature, the early estimation of performance parameters for web services is not addressed. In this paper, a methodology to estimate the performance metrics, such as average response time, average service time, average waiting time, and probability of idle server and dropping of sessions is proposed considering the dynamic workload of web services. Furthermore, the methodology provides an opportunity to determine the performance goal.

3 Methodology

The performance of web services highly depends on the configuration of the resources and service requirements of various services and in particular, the demand for them. Hence, precise performance analysis is necessary to estimate the expected demand for the web services and in turn for the corresponding servers. It is acknowledged that the workload of a web service in a time horizon is particularly important for identifying the capacity of the deployment environment and the allocation of web services to web servers. Subsequently, a probabilistic technique is necessary to find out the workload in a fixed time horizon. This type of analysis in a fixed time horizon helps to find out the adequacy of web servers and application servers and also to assess the alternatives in the deployment environment so that the performance goal can be achieved.

Fluctuations in workload may have an impact on the performance due to contention for resources during the execution of the web services. Moreover, it may lead to 'drop the requests' also. Hence, an estimation mechanism has to be provided to determine the utilization of the web servers. The estimation is based on the design of the deployment environment and the workload during the time horizon. Suitable configuration for the servers can be determined by analyzing the deployment environment considering different configurations for the servers.

Considering all the above aspects, a methodology is proposed in this paper for estimating the performance of service-oriented web services over a time horizon considering aspects of WA/WS. The methodology aims to identify whether the proposed configuration of the deployment environment of web servers is adequate to satisfy the customers based on the required demand or not.

Let $i \in [i_0, I+ i_0-1]$ be the interval, where i_0 be the first interval in the time horizon that is chosen; *I* will be the number of intervals. Let A_T, be the set of activities that are expected to be processed during the 'I' intervals and each activity may trigger a particular web service of a composite service. Based on these assumptions, the methodology has been devised as follows.

- Modeling the workload of web service components using a mathematical model over a given time horizon.
- Modeling the servers and other hardware resources in the deployment environment of WS architecture in SOA.
- Formulating a procedure to calculate the response time proportional to workload.
- Prediction of performance and analyzing the behavior of

the resources across the servers of web services.

- Identifying bottleneck resources and improving the performance by carrying out sensitivity analysis on resources of composite web services.

3.1 Modeling the Workload of Composite Web Services

A workload element is a unique workload request that is generated for a given web service and must be handled by the appropriate composite web server. The workload specification covers service utilization or requests for service functions, as well as the likelihood of requests arriving and request patterns [1]. The activity diagram of Unified Modeling Language (UML) helps to model the flow of activities of the web services, and it is available during the preliminary design phase. A sequence of actions involved in the activity is the scenario of a service. The activity scenarios define the desired behavior of web service and show the execution patterns of a composite service. A scenario of service illustrates the interaction between the objects or execution of the activities at a particular time. The execution of a specific activity scenario depends on the type of user's request (event). Web services are geographically distributed, and the number of users of the system tends to vary from time to time (for example, the travel agent service, derived from [24], is a good example of how this works. A user accesses a single interface, entering the information needed to book a flight, book a hotel room, and obtain maps from the area. Each of these three obligations will be met by its web service. The problem here is that the user may submit some information that only needs to be used by certain parts of the service. For instance, a credit card is involved with airline and hotel reservations, but a website can provide maps for free. As a travel agency customer, one would want to know whether the credit card information will be secured and accessible to only necessary vendors). As a result, in a service-oriented environment, the type of requests arriving at a particular time interval is unpredictable. Hence, workload scenarios can be built based on the number and type of expected activities across a certain time horizon interval.

Let F_I be the collection of activities that represent the web service functionality that will occur throughout I time intervals, and R_i denotes the number of requests that can arrive in the interval $i \in [i_0, I+ i_0-1]$.

Each request is characterized by:

- $p_{i,r}$– the likelihood of the request r occurring at a given time interval i
- $D^{a}_{i,r}$ – expected demand for the activity $a \in F_I$, if the request r arrives among those specified at the time interval i

The arrival of the request at period i and the scenario that occurred in period i-1 are used to define the workload of a certain service of a composite web service at period i. Let S_i denote the number of demand scenarios for a service that will occur in each interval i, with $p_{i,s}$ denoting the likelihood that scenario s will occur at interval *i*.

During the first interval i_0, the number of scenarios can be calculated as:

$$S_{i_0} = R_{i_0} \tag{3.1}$$

The number of scenarios in the following interval $i \in [i_0, I+ i_0-1]$. can be computed recursively as:

$$S_i = R_i \,.\, S_{i}\text{-}1 \tag{3.2}$$

The probable situations at a time interval i depend on the scenarios at a time interval i-1, and the requests come at the time interval i for a certain composite web service.

The conditional probability tree is used to depict workload scenarios in the service-oriented environment because the occurrence of a service scenario at any time interval conditionally depends on the preceding scenario and the arrival of requests in that scenario. Figure 1 depicts the layout of the workload scenarios.

Let $p_{i,s}$ be the probability that the scenario of a service *s* can be utilized in the period *i*, then

$$p_{i,s} = p_{i,r}.p_{i-1,v} \tag{3.3}$$

where, $p_{i,r}$ be the probability that the request r occurs in the period *i* and $p_{i-1,v}$ be the probability that the scenario *v* occurs in the period *i-1*. The demand $D^{a}_{i,s}$ is computed as:

$$D^{a}_{i,s} = p_{i,s}\forall \tag{3.4}$$

Where $D^{a}_{i,s}$ is the demand for activity a in time interval t respond to the workload scenario s.

Furthermore, the state (activity) at the interval *i-1* and the request arrived at *i* determine the state of the composite service in time *i*. Hence, the group of activities to be executed at time *t* are considered as state of the web service at the time *i*, and the pattern of processing the activities are modeled as the State Chart Diagram of UML. The representation of the workload scenarios is presented as a statechart diagram in Figure 2.

3.2 Modeling of Execution Environment of a Composite Web Service

The resources in the execution environment and workloads are closely related to each other. The configuration of the resources available for individual web services as well as the workload for the web services influences the performance of the composite web service.

The software architecture that includes hardware and software components becomes the base for the deployment environment of each web service. The features of the execution environment are specified by a set of attributes that categorize each resource in the environment. For analyzing the deployment environment, a different set of attributes for resources can be considered to obtain the performance metrics of those resources.

For each hardware resource, the following set of attributes are considered:

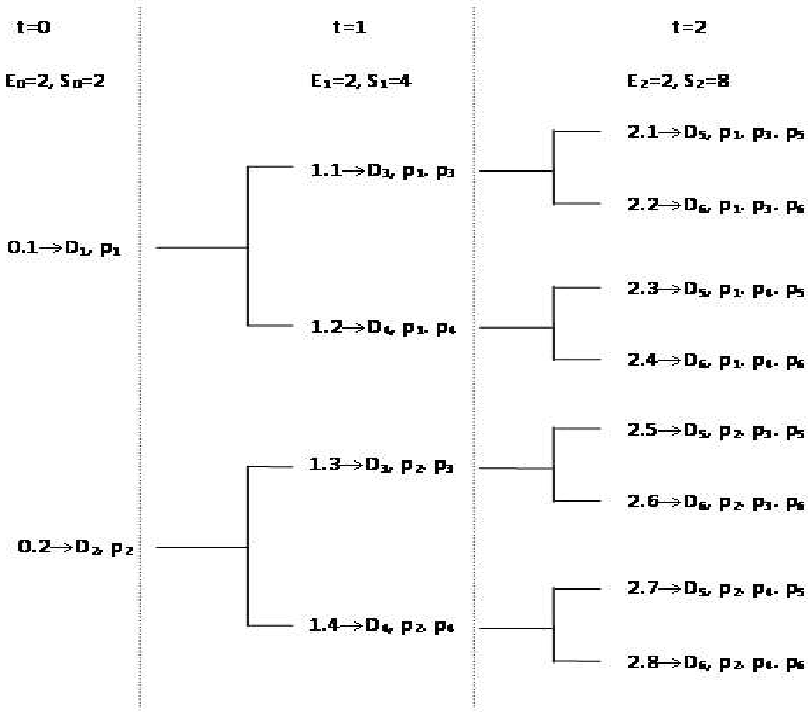

Figure 1: Workload scenarios – conditional probability

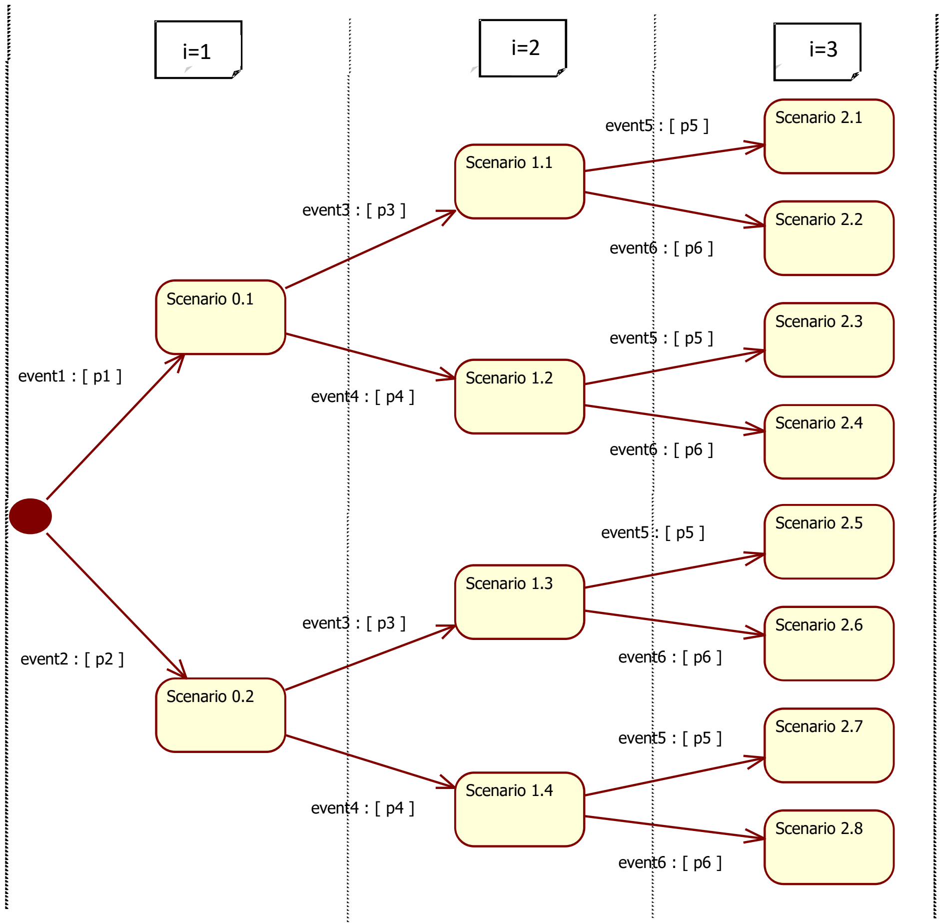

Figure 2: Workload Scenario - Statechart Diagram

- The configuration
- A collection of activities belonging to F_I that the resource can carry out
- The number of resources used by each activity

Based on the resource parameters that are to be specified, a set of values can be assigned to define the base configuration for the resources. For example, consider a resource R_1 with the configuration of 1000 KB/sec as the processing speed of a web service. Let a_1, a_2, a_3, be the activities that require the service of the resource R_1 and the resource usage for the activities be 5 seconds, 8 seconds, and 4 seconds, respectively. Another alternative type of data can be obtained by changing any of the combinations of these attributes; for example, the processing speed changing to 10000 KB/sec or the processing time for the activities to 2 seconds, 8 seconds, and 5 seconds, respectively.

3.3 Calculation of Response Time

The resource utilization of each web service must be analyzed to determine the composite web service's optimal and predefined response time over the provided time horizon. This can be made possible by calculating the resource utilization for each time interval. The configuration and the fluctuating workload scenarios lead to a difference in resource usage so that each configuration $c \in IC$ is characterized by a unique set of parameters of the resources of that web service.

Let PT_a^j be the processing time required for the activity, $a \in F_I$ for a given configuration *j*. Let $T_{i,s}^j$ be the total processing time in time interval *i* to respond to the workload scenarios *s*. Then the total processing time during the period *i* can be calculated as:

$$T_{t,s}^{j} = \sum_{a \in F_I} \left(D_{i,s}^{a} * PT_a^{j} \right) \quad (3.5)$$

4 Illustration of the Methodology

For a precise illustration of the methodology, we have taken a case study on the Travel Agent web application/web service [13]. Users can use the Travel agency's website to search for available airlines, hotels, and cars that suit their search parameters, make reservations, and pay for booked services, among other things. The Travel Agency web application makes use of a local database that stores customer information as well as tourism-related data. Moreover, four external web services are used by the Travel Agency web application: the airline web service provider which provides services like flight availability, the hotel online service provider publishes a news bulletin and accepts room reservations, the vehicle web service provider conveys availability and booking information, and an independent online payment web service provider collects payments. Not only does the Travel Agency use web services, but it also exposes some of its functionality as a web service. The UML models developed for the Travel Agency web application are given in the following sections.

4.1 Modeling the Scenarios of Workload

The kind of queries that the application receives are determined by its functionalities. The overall activity of the travel care application is presented in Figure 3 with a help of a use case diagram.

Figure 4 depicts the flow of activities carried out in the application for any type of reservation made through travel care. The action begins with a login to the travel care system using the credentials created and a booking query (airline, car, and hotel). Once the type of reservation is chosen, the travel care selects the service providers and connects to the appropriate network, then completes the reservation by selecting and providing all of the necessary details, confirms the transaction by making an online payment, and ends the transaction by reconnecting the network.

The use case diagram of the Airline Reservation is shown in Figure 5 with various scenarios. The scenarios or services that are frequently used by the customers are Login for a particular travel site for reservation, Searching a flight, Seat selection, Providing booking details, Confirming reservation, Canceling a reservation, Payments, etc.

The activity diagram shown in Figure 6 describes the flow of actions carried out by a customer who reserves a flight ticket. The activity begins with login in into the travel agency by entering the credentials. Based on the destination planned, the flights are searched and selected by reserving the class, number of seats, and providing the passenger details. On confirmation of the reservation, the payment is made based on the seat count and the transaction is ended by generating the bill for the payment.

The use case diagram for Car Reservation is shown in Figure 7 that includes the following use cases: Registering and login for a particular travel reservation, selecting the location of car provider, checking the availability of car based on the particulars, selecting the car by providing details such as rent date and time, duration and number of cars, confirming and cancelling the booking, making payment on confirmation.

The activity diagram for Car Reservation is presented in Figure 8 which starts by selecting the location of the car providers nearer to the customer's location. The customer then checks for the availability of cars by providing certain particulars. If available, the customer reserves a particular type and number of cars. The customer confirms the reservation and provides personal details, makes the payment, and terminates the reservation process.

The use case diagram of the Hotel reservation is shown in Figure 9. It includes the following functionalities: Registering and login into a travel agency, selecting a location for a customer's choice, selecting the hotel in the location and checking for availability of rooms, selecting the type and number of rooms, confirming and cancelling booking, making payment. The actors include travel agencies, hotel employees, banks, and customers.

The hotel reservation process is presented in Figure 10, which starts with the selection of the location of customers' choice by viewing the map. The hotel in the selected location is checked for the availability of rooms. If available, the customer selects the room type and room count and then proceeds to book by providing check-in, check-out dates, and personal details. The booking is completed by making an online payment for the reserved room.

The conditional probability tree and the statechart diagram for Travel Care, Airline Reservation, Hotel Reservation, and

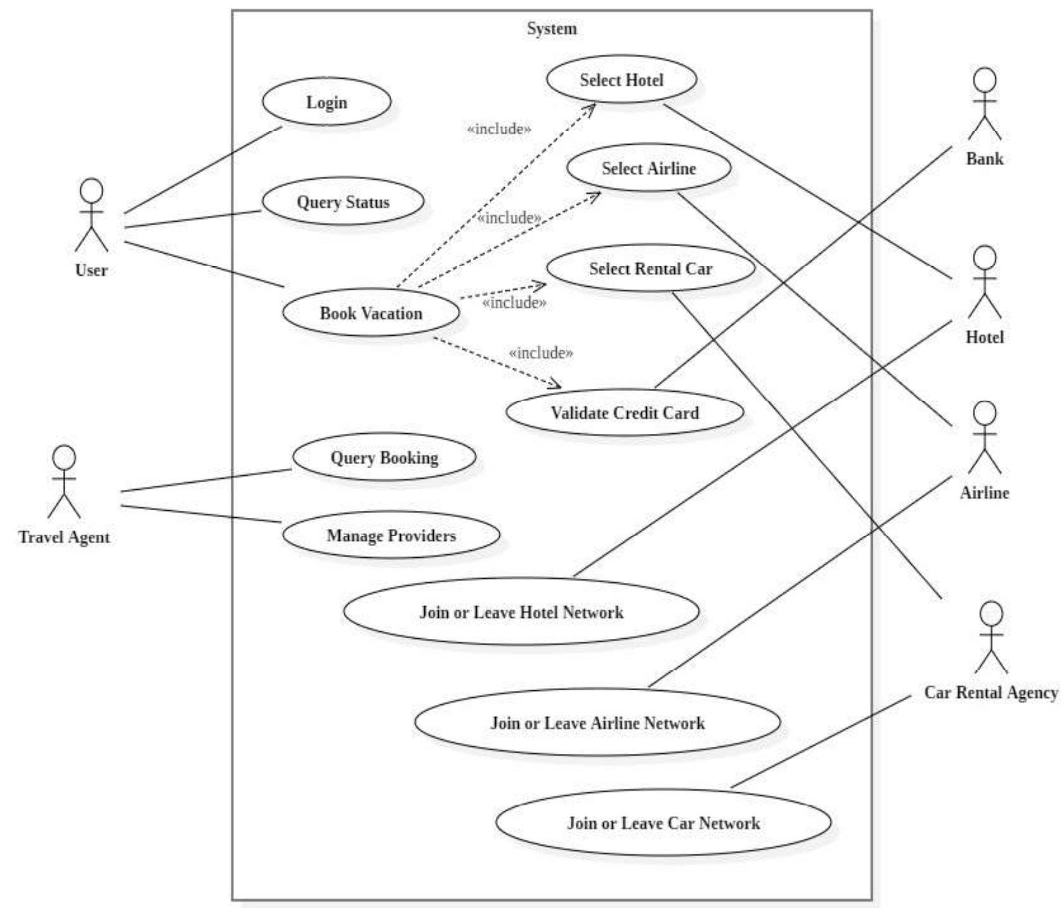

Figure 3: Use case model of travel care

Car Rental are given in Figures 11 to 15. These diagrams are used to represent the expected scenario of activities during each of the time intervals. The occurrence of these activity scenarios is dependent on customers' requests during the specified time interval and the probability of their occurrence.

4.2 Modeling the Execution Environment

The deployment environment of the Travel Care is given in Figure 16. The execution environment of the application is modeled using the deployment diagram of UML. The data for the elements of execution are given in Table 1.

4.3 Response Time Calculation

The proposed model is simulated by considering ten-time intervals, for a given time horizon t, t∈[10, 12]. To calculate the total processing time for the scenarios, the following information is needed as given in equation 3.5: i) Probability of the activity occurring in a given time interval t. ii) the processing time of each activity. Uniform distribution is used to generate the probability of occurrence of the activities. The execution time of the activities is estimated using the methodology discussed in [17]. The activity point performance prediction approach is used to calculate the activity points in this methodology. The size of the activity is calculated in terms of Lines of Code (LoC) using the gearing factor and in turn activity size in kilobytes is obtained from LoC. The processing time for the activities is calculated from the software size taking into consideration the given deployment environment. The estimated response time for the activities is calculated and tabulated in Table 2. The processing speed of the hardware resources is tabulated in Table 1.

For example, the probability of the occurrence of activities considered during the time interval t_1 is 0.9. To process the activities that are to be handled during this time interval, the time required is estimated as 0.355 seconds. Then during the time interval t_1, the time required for processing this scenario can be computed by applying Equation 3.5 as:

$$T_{t,s}^{j} = 0.9 * 0.355 = 0.32 \sec$$

The maximum, minimum, mean and execution time, variance, and standard deviation are calculated and tabulated in Table 3 for each of ten intervals (i_1, i_2, i_3 … i_{10}), as well as for the total time horizon (T* - the sum of response time acquired in intervals i_1 to i_{10}). Smith [18-19] provides the following reasons for estimating the minimum, maximum, and average response times:

i. It aids in the study of the best-average-worst case analysis.

Figure 4: Activity model of travel care

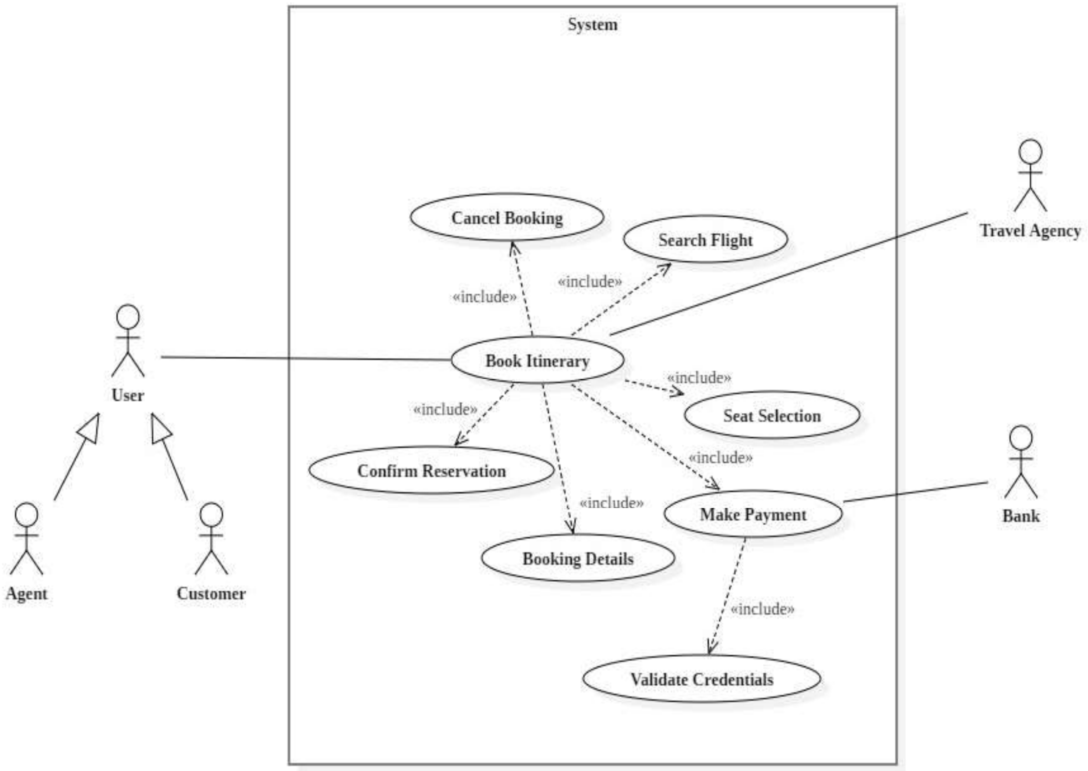

Figure 5: Use case model of airline reservation

ii. The obtained maximum response time in various trials aids in determining the application's performance goal

As shown in Figure 17, a graph is generated to display the response time acquired for ten intervals. Moreover, for the response time obtained during time intervals i_1 to i_{10}, graphs are generated and presented in the figures, from Figure 18 to Figure 27.

The observations are:
The maximum response time in interval t_1 is less related to other intervals. The only reason for this is login-related actions; user authentication can happen during this time. Furthermore, these activities are the interactions between the user and the travel reservation. The variance and the mean response time are comparably higher in the intervals t_4 to t_7, because the activities that occur during this interval need more execution time. This is due to the activities of the interactions between the travel reservation and the application servers, namely, airlines, car rental, and hotel reservation; these interactions are communicated through protocols service. The total maximum response time is obtained as 5.721 seconds, for the intervals considered.

4.4 Simulation Results

Simulation is carried out with 500 trials, where 30000 data that represent requests for web service are considered for each trial. Uniform distribution is used to generate the probability of occurrence of the requests. The maximum response time is obtained in the range of 4.73 to 5.82 seconds irrespective of the workload during the intervals t_1 to t_{10}. Moreover, a negligible difference is observed in the mean response time value. Graphs are generated for understanding the fluctuations visually. Sample graphs are presented in Figures 17, 18, and 19 for intervals 1, 3, and 4, respectively.

The preceding observations lead to a conclusion that irrespective of the workload during the given time horizon consisting of ten-time intervals, the required processing time for executing the activities is 5.82 seconds. This is the maximum time taken for the given configuration in the deployment environment of web services.

The recommendations that can be suggested are:

- The response time obtained as the maximum can be considered as the minimum required processing time to process the activities of the web service during a heavy workload since we have not considered the congestion delay. Therefore, this maximum response time value with probable congestion delay may help to define the performance goal of a composite web service.
- While a huge number of users using the system, a user cannot expect to receive the response within 5.82 seconds.
- If the performance objective is 5.82 seconds or above, then the configuration of the resources in the execution environment can be chosen as the values given in Table 1 (configuration C_1).

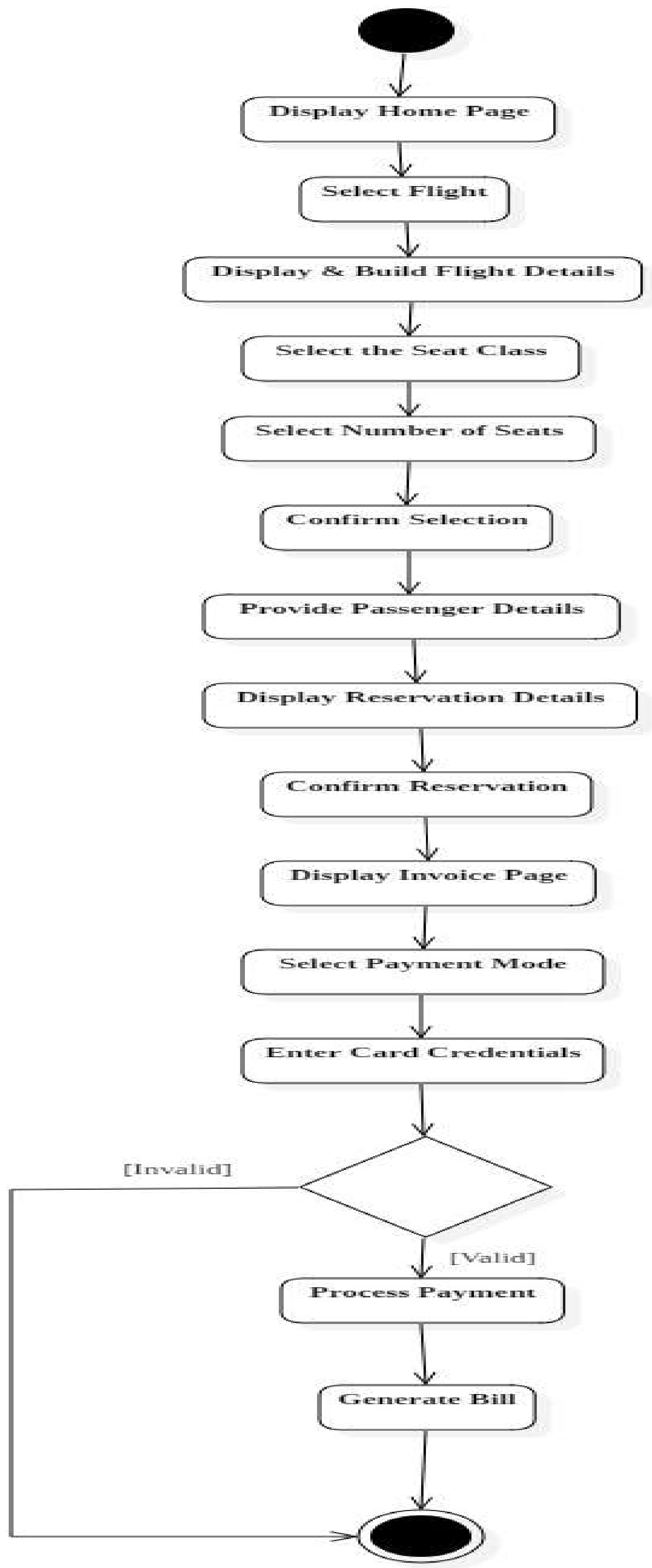

Figure 6: Activity model of airline reservation

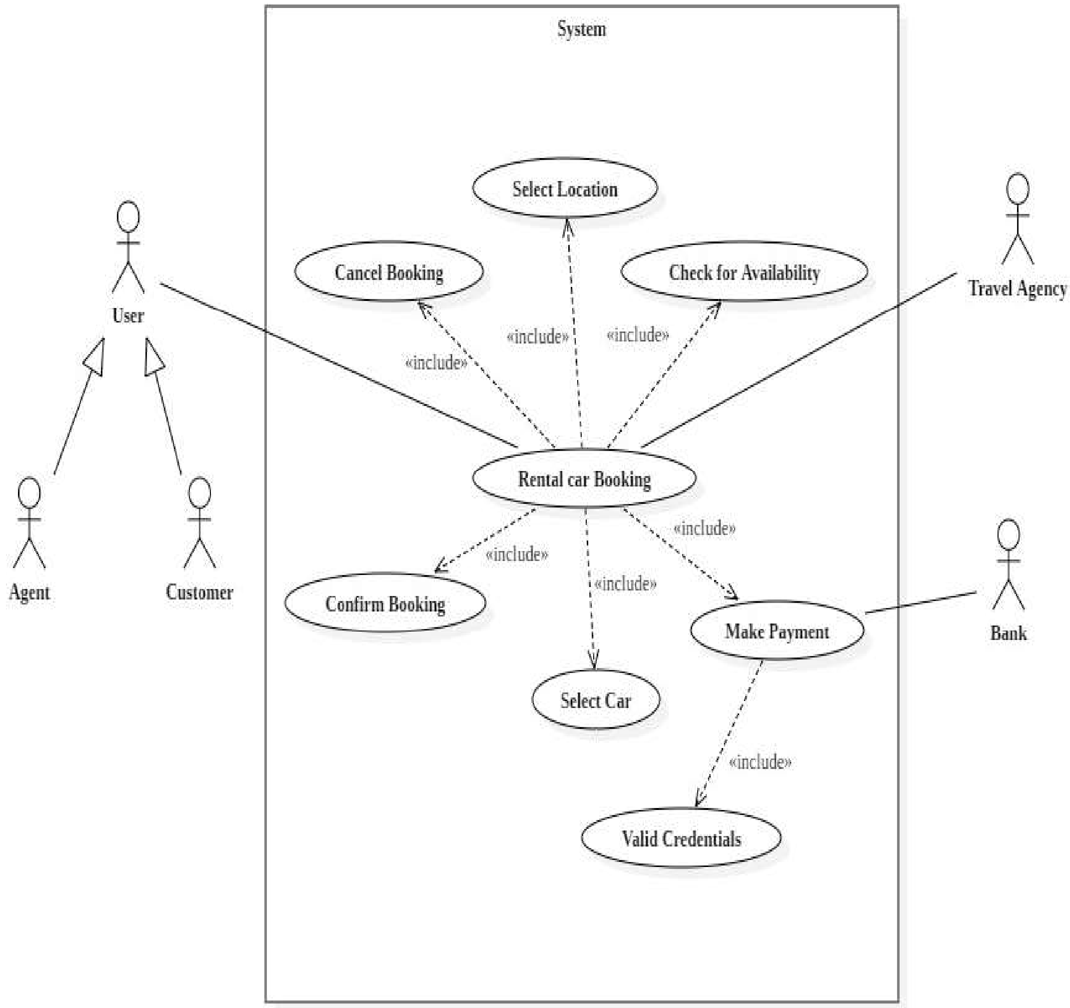

Figure 7: Use case model of car reservation

Section 7.6 discusses the analysis of changes in the configuration of hardware resources in the execution environment, as well as the identification of bottleneck resources and ideas for improvement.

4.5 Sensitivity Analysis

The environment of the composite web services for Travel Reservation is simulated using the simulation tool SMTQA (Simulation of Multi-Tier Queuing Applications) [6]. The simulation is carried out with the configuration (C1) given in Table 1, and the performance metrics are obtained. The values of the performance metrics are given in Table 4. It is observed from the table that the probability of dropping of sessions in Internet 1 and Internet 2 is 0.699 and 0.497, respectively. This is due to the low processing speed of the Internet. Hence, these are identified as bottleneck resources. To analyze the behavior of the hardware resources in the execution environment, sensitivity analysis is carried out by considering a modification in resource configuration one at a time. To improve the performance of the services, the processing speed of the Internet 1 is increased to 1050 KB. As a consequence of this, the probability of dropping requests is reduced to 0.213.

Since the number of requests processed by the Internet 1 is increased, this has an impact on the performance of the Travel Reservation server. As a consequence, the dropping of requests has happened in the Travel Reservation server, and its probability is 0.225. In turn, the probability of dropping sessions on the Internet is reduced from 0.497 to 0.297. From this observation, we could conclude that the performance of the Internet 1 and Internet 2 can be improved further by increasing their processing speed. Simultaneously, the processing speed of the Travel Reservation server also must be increased to avoid the dropping of requests.

5 Conclusions

The given methodology provides a mathematical model to estimate the response time based on the workload that fluctuates over a given time horizon. The estimation is made by i) Modeling the servers and other hardware resources in the deployment environment of WS architecture in SOA. ii) Formulating a procedure to calculate the response time pro-

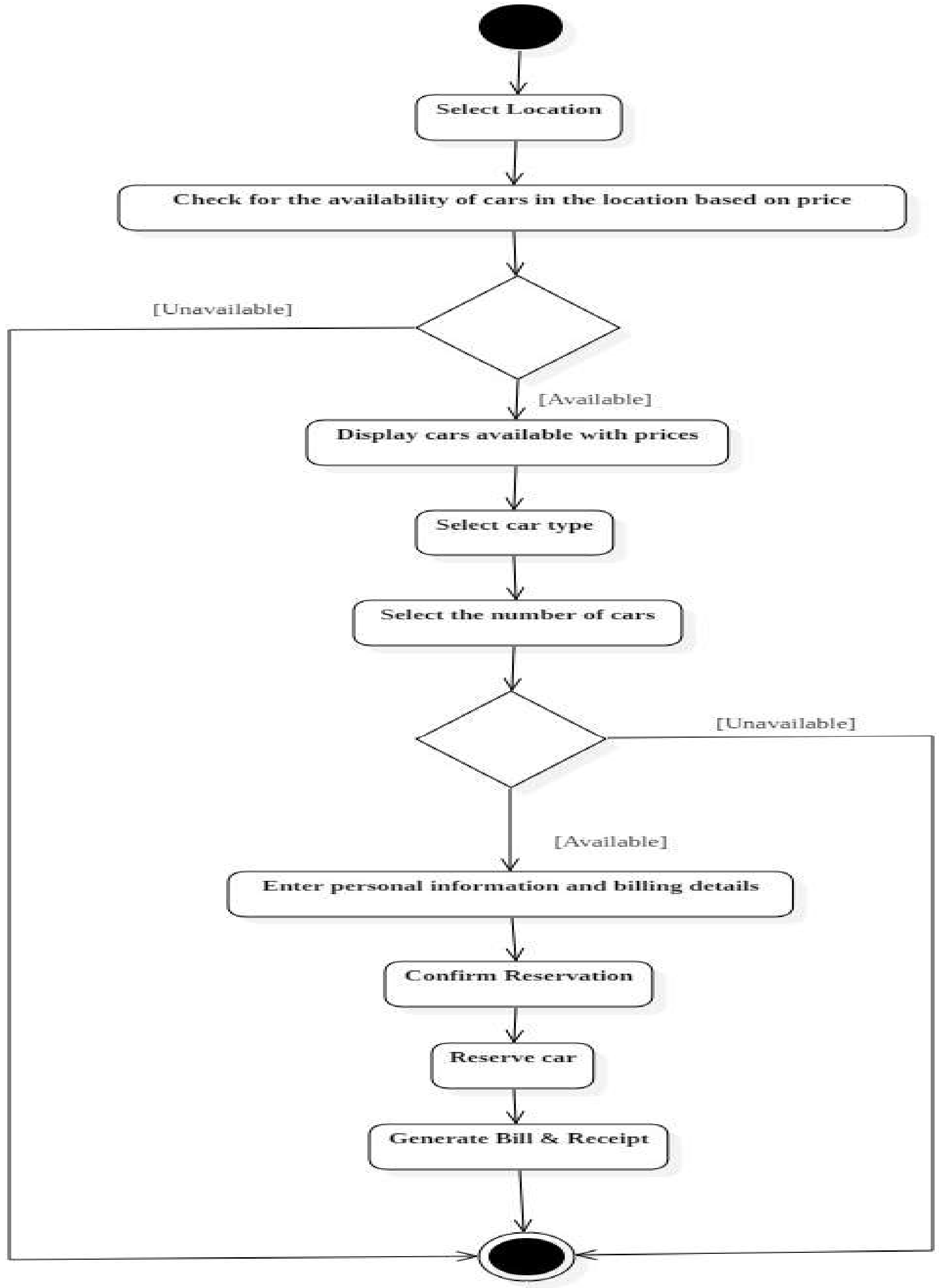

Figure 8: Activity model of car reservation

portional to workload. iii) Prediction of performance and analyzing the behavior of the resources across various web servers. iv) Identifying bottleneck resources and improving the performance of composite web service by sensitivity analysis.

Sensitivity analysis using the tool SMTQA is carried out to analyze the behavior of the hardware resources. The sensitivity analysis has indicated how changes in the base configuration of resources have an impact on the response time of the application. The methodology also helps to determine and define the performance objective of the web services by obtaining the range of values for maximum response time.

The proposed methodology can be used to predict the performance and to determine the most suitable deployment environment that can achieve the defined performance objective for web service based on non-uniform workload considering specifications of WS architecture. But, in composite web service various other services may be affecting the response time of the application; hence, the workloads of other services also need to be considered.

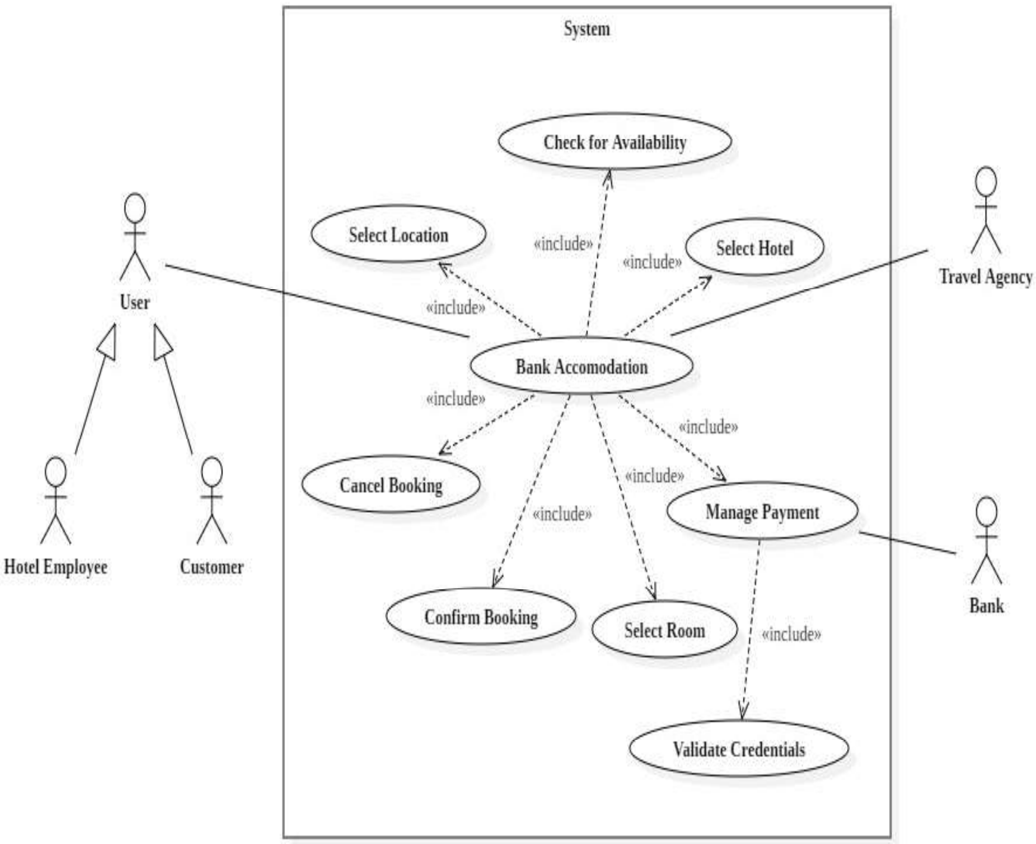

Figure 9: Use case model of hotel reservation

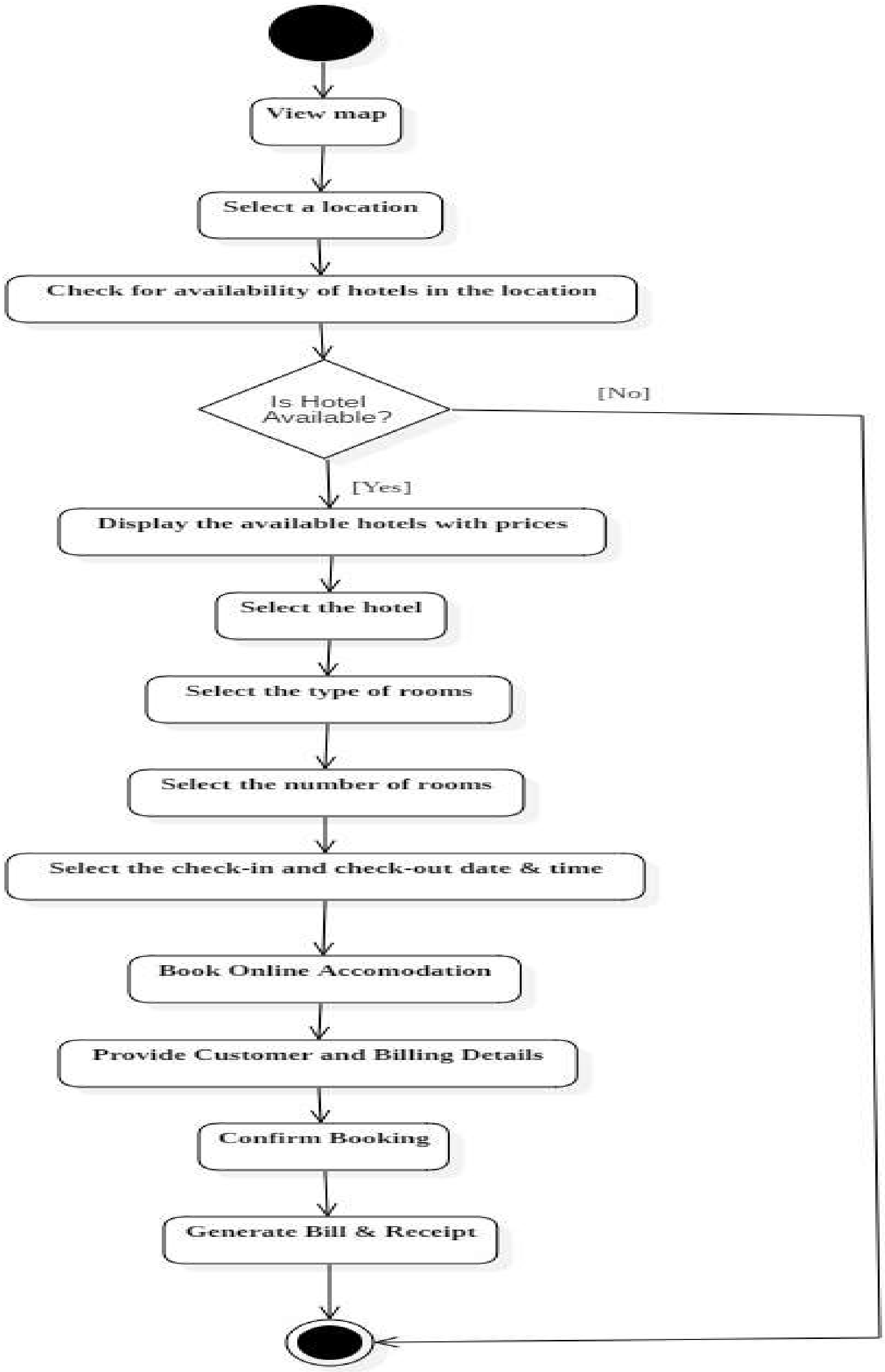

Figure 10: Activity model of hotel reservation

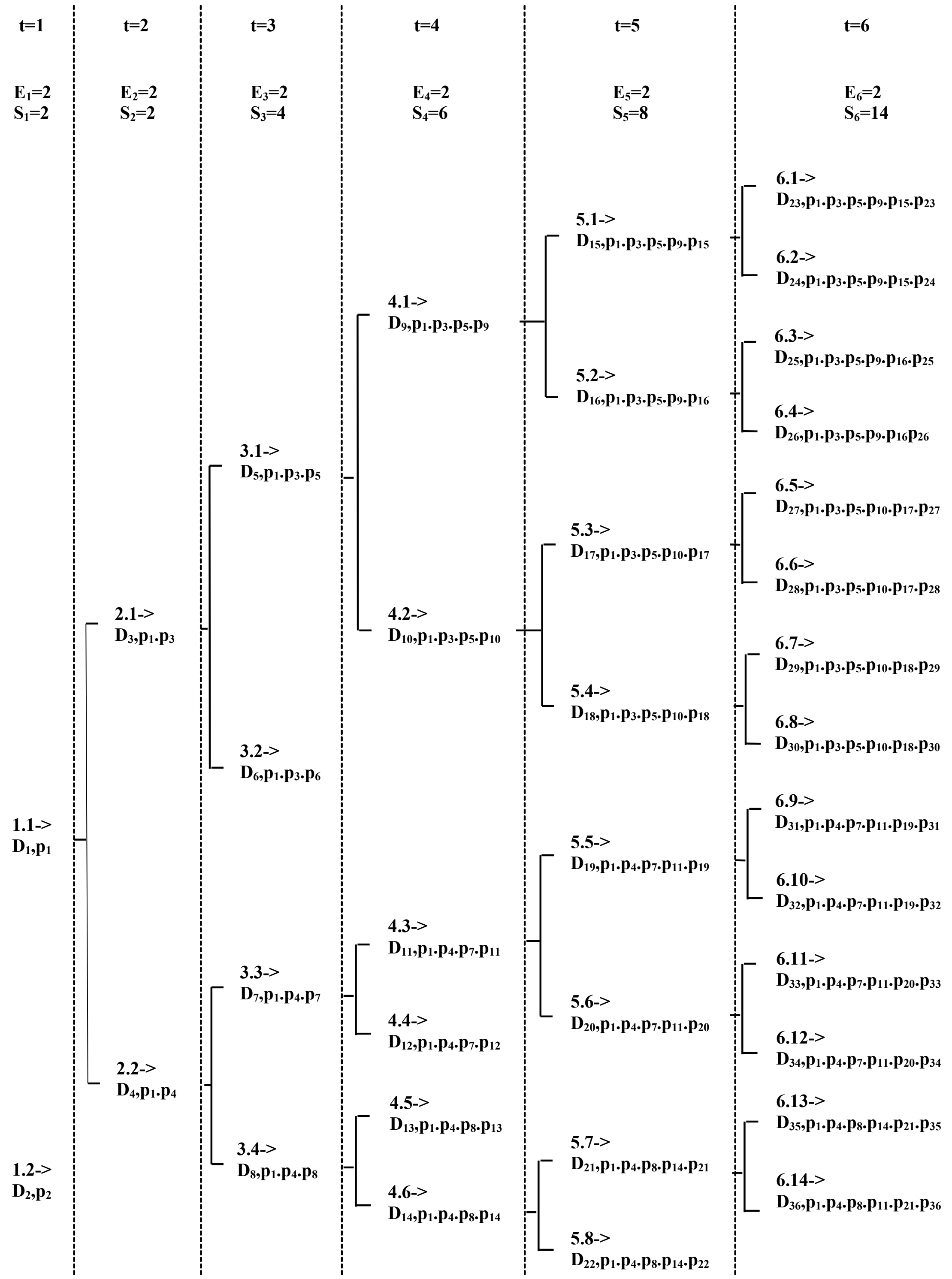

Figure 11: Conditional probability tree for the case study

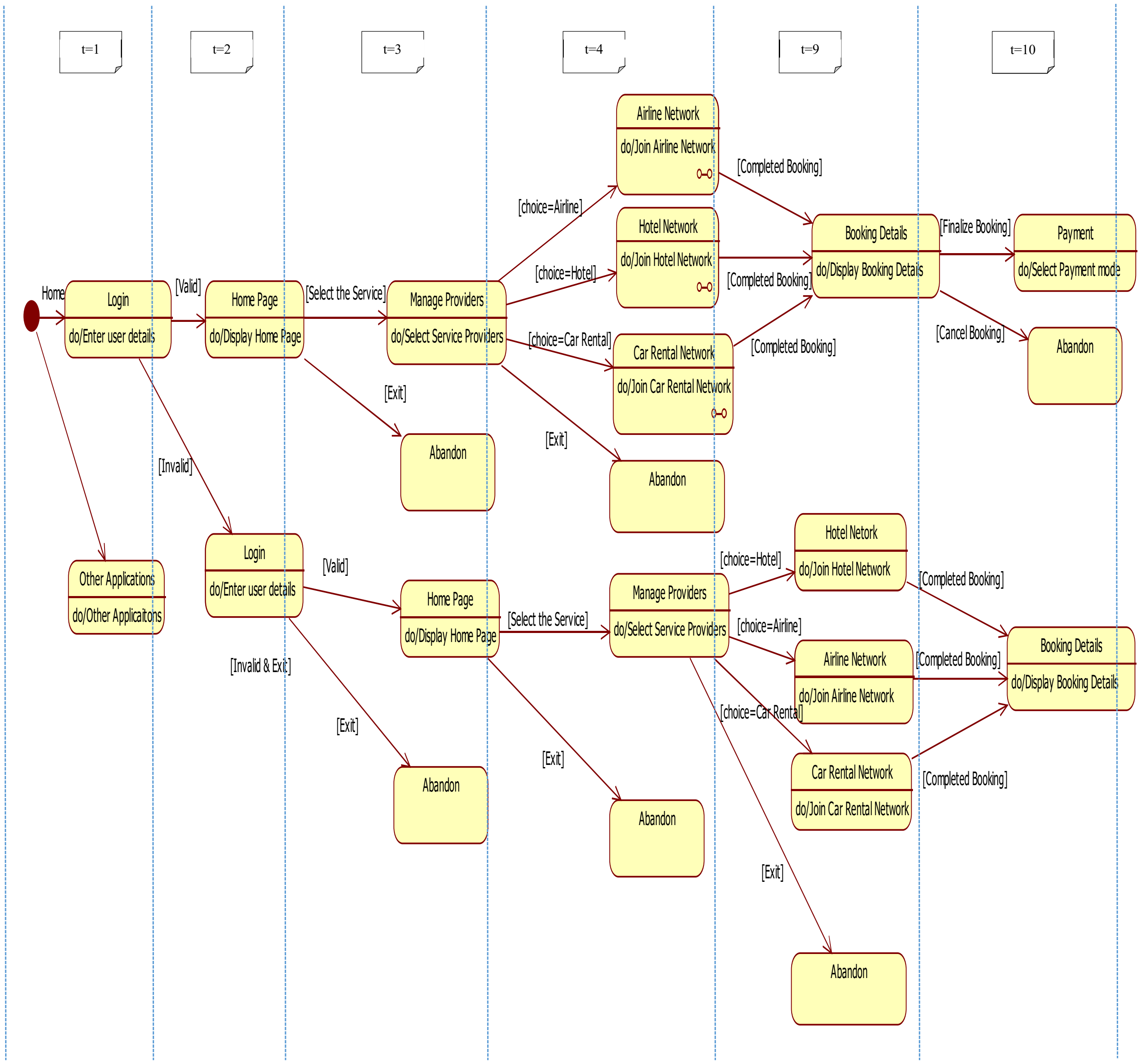

Figure 12: State chart diagram for travel care

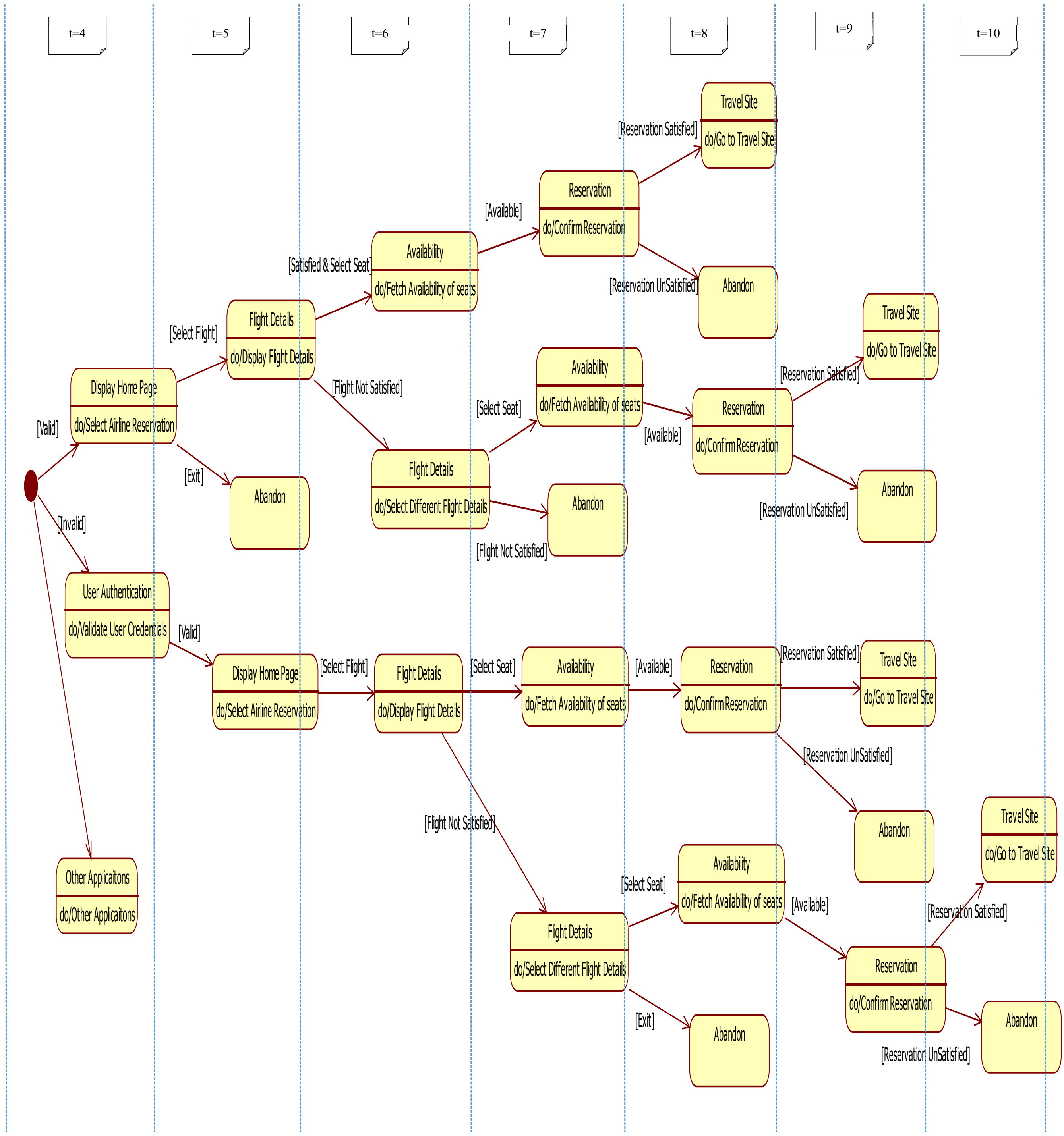

Figure 13: State chart diagram for airline reservation

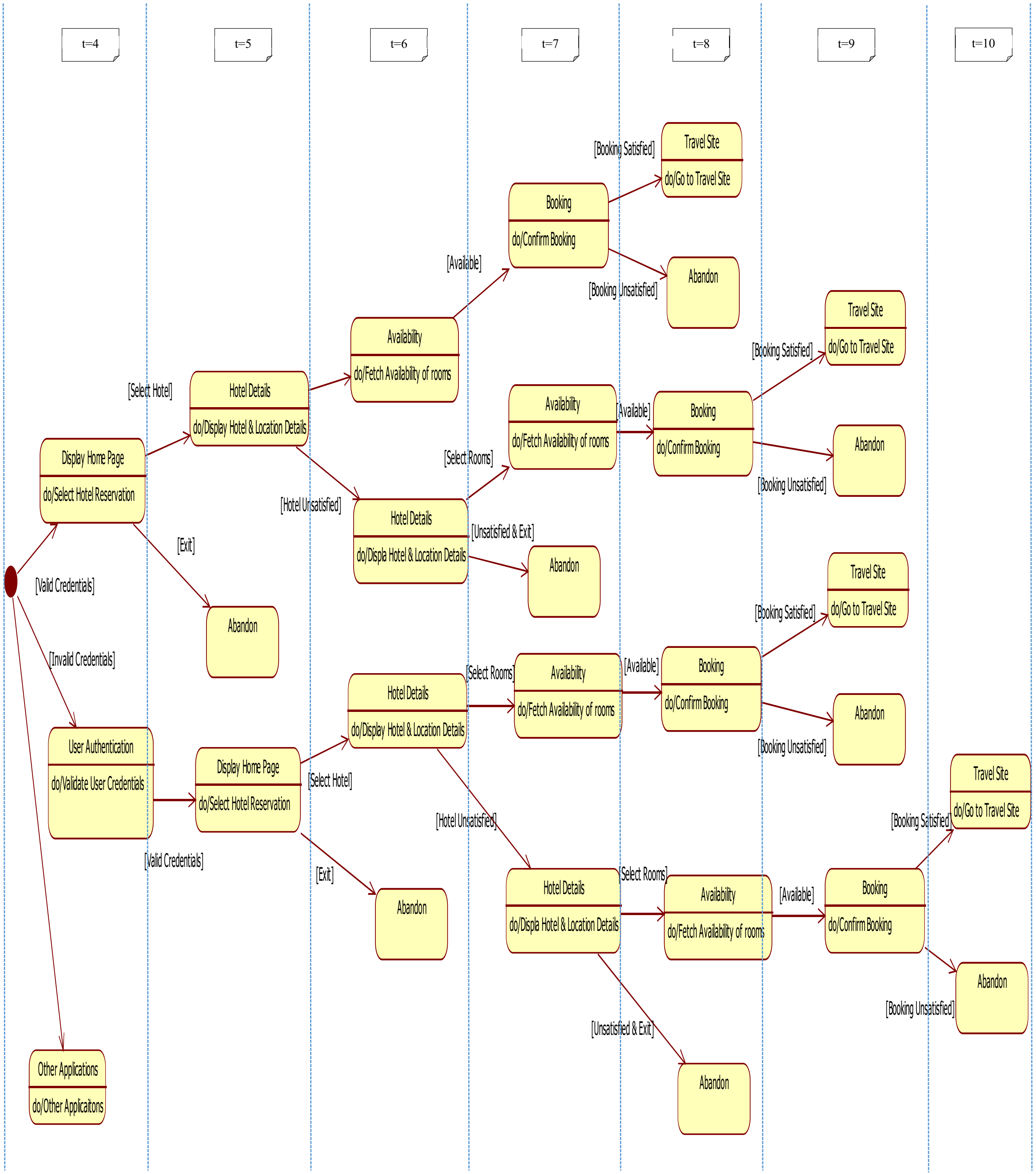

Figure 14: State chart diagram for hotel reservation

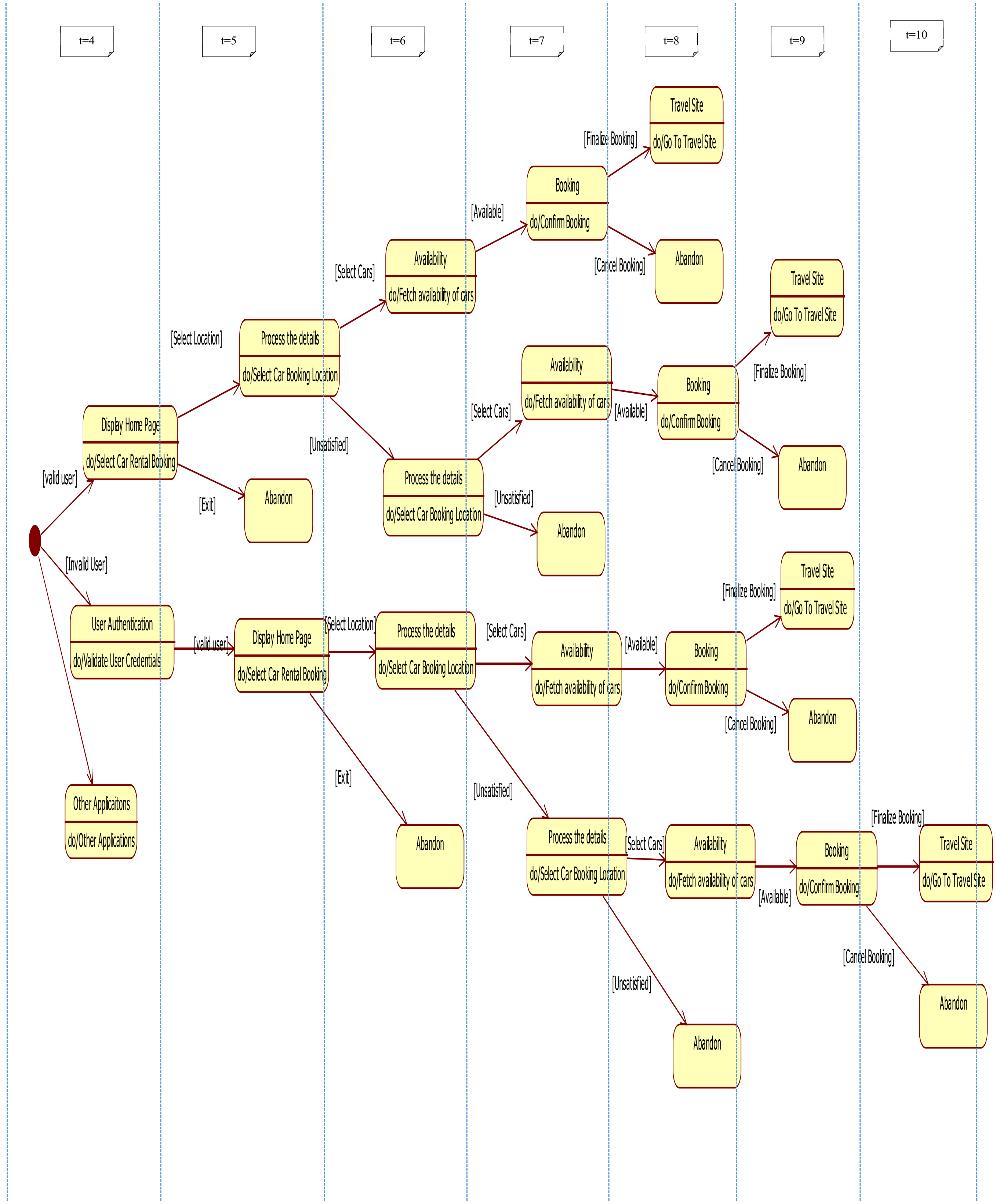

Figure 15: State chart diagram for car reservation

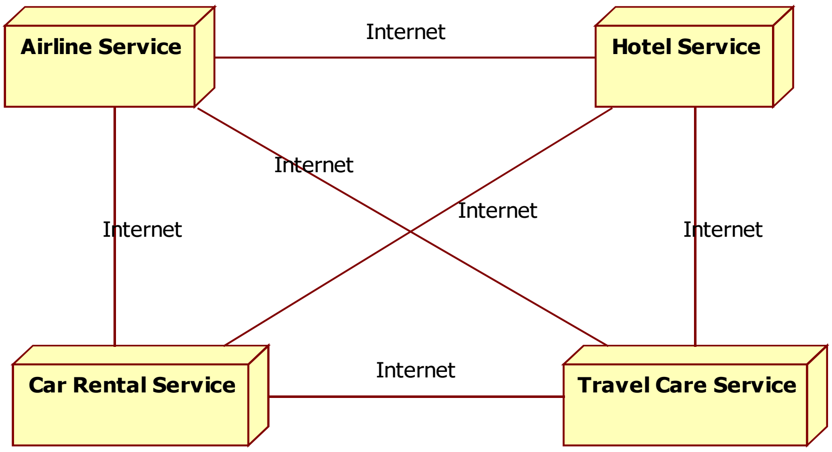

Figure 16: Exécution environment – base configuration (C_1)

Table 1: Configuration of resources in the execution environment (C_1)

Devices / Servers	*Client*	*Internet*	*Airline*	*Hotel*	*Car*	*Credit Card*	*LAN*
Processing Speed (Sec/KB)	16000	575	20000	15000	15000	20000	12500

Table 2: Size and response time of activities of travel service

Travel Care Service		
Activities	**Size (KB)**	**Response Time (Sec)**
Login	81.83094	0.355473
Query Status	55.6554378	0.472115
Select Airline	61.8863298	0.436247
Select Hotel	61.8863298	0.436247
Select Rental car	61.8863298	0.436247
Query Booking	53.8247277	0.421855
Manage Providers	78.27246088	0.465031
Join/leave airline net	95.1955894	0.543976
Join/leave hotel net	95.1955894	0.543976
Join/leave car net	95.1955894	0.543976

Table 3: Statistical values for configuration

Configuration											
	i1	**i2**	**i3**	**i4**	**i5**	**i6**	**i7**	**i8**	**i9**	**i10**	**Total**
Max	0.350	0.369	0.438	1.109	1.023	1.088	1.042	0.638	0.484	0.376	5.721
Min	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.005
Mean	0.174	0.099	0.066	0.170	0.093	0.077	0.169	0.086	0.041	0.019	0.993
Variance	0.010	0.006	0.005	0.028	0.014	0.010	0.015	0.006	0.002	0.001	0.429

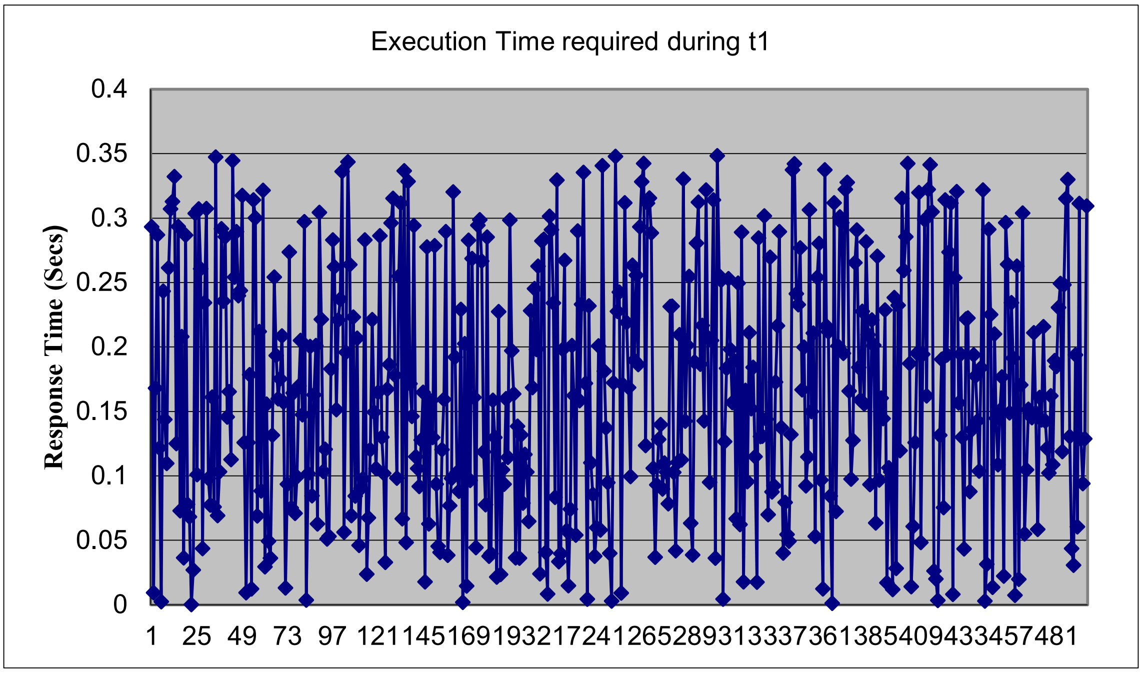

Figure 17: Response time obtained for t1

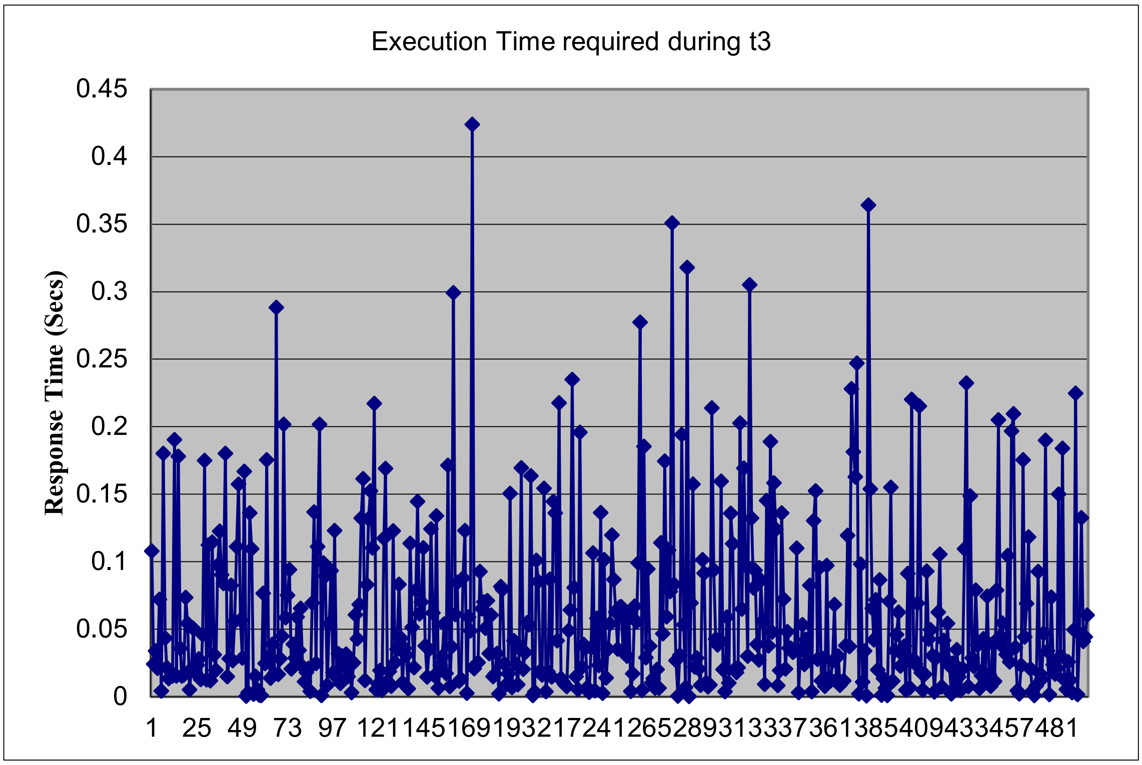

Figure 18: Response time obtained for t3

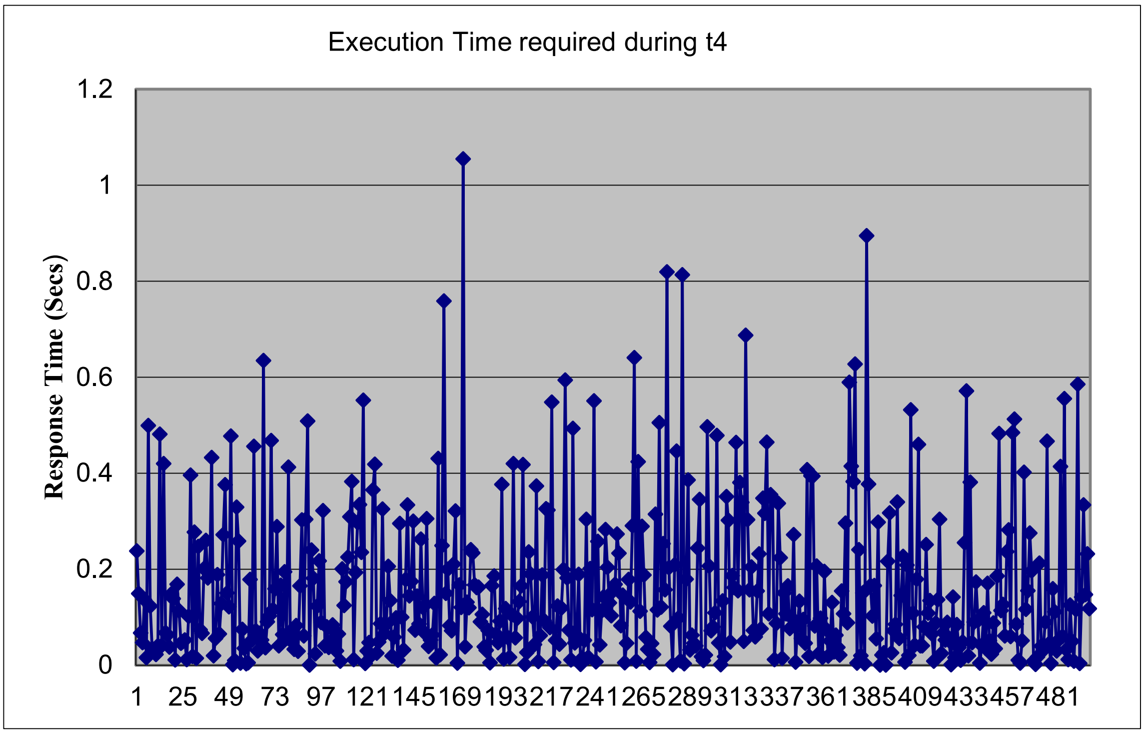

Figure 19: Response time obtained for t4

Table 4: Performance metrics of Internet speed 575 KBps with Arrival distribution is 0.01

Sl No	Layer Name	Processing Speed (Units)	Average Response time	Average service time	Average waiting time	Probability of idle server	Probability of dropping sessions
1	Client	16000	0.004	0.003	0.000	0.764	0.003
2	Internet1	575	0.144	0.052	0.091	0.001	0.699
3	Travel Reservation	20000	0.001	0.001	0.000	0.990	0.000
4	Internet2	575	0.090	0.036	0.055	0.011	0.497
5	Airline	20000	0.001	0.001	0.000	0.976	0.000
6	Hotel	15000	0.002	0.002	0.000	0.963	0.000
7	Car	15000	0.002	0.002	0.000	0.977	0.000
8	CreditCard	20000	0.004	0.004	0.000	0.999	0.000
9	LAN11	12500	0.003	0.003	0.000	0.989	0.000
10	LAn12	12500	0.003	0.003	0.000	0.984	0.000
11	LAN13	12500	0.002	0.002	0.000	0.995	0.000
12	LAN14	12500	0.004	0.004	0.000	0.997	0.000
13	WS1	10000	0.004	0.004	0.000	0.985	0.000
14	WS2	10000	0.004	0.004	0.000	0.981	0.000
15	WS3	10000	0.003	0.003	0.000	0.993	0.000
16	WS4	10000	0.005	0.005	0.000	0.997	0.000
17	LAN21	12500	0.003	0.003	0.000	0.983	0.000
18	LAN22	12500	0.002	0.002	0.000	0.978	0.000
19	LAN23	12500	0.002	0.002	0.000	0.993	0.000
20	LAN24	12500	0.003	0.003	0.000	0.998	0.000
21	DB1	20000	0.001	0.001	0.000	0.997	0.000
22	DB2	15000	0.001	0.001	0.000	0.996	0.000
23	DB3	15000	0.001	0.001	0.000	0.998	0.000
24	DB4	20000	0.004	0.004	0.000	0.999	0.000

References

[1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, *Web Services – Concepts, Architectures, and Applications*, Springer-Verlag, Berlin, Germany, pp. 123-149, 2004, doi:10.1007/978-3-662-10876-5_5.

[2] V. Almeida and D. Menasce, "Capacity Planning an Essential Tool for Managing Web Services," *IT Professional*, 4(4):33-38, July 2002, https://doi.org/10.1109/mitp.2002.1046642

[3] C. Catley, D. C. Petriu, and M. Frize, "Software Performance Engineering of a Web Service-Based Clinical Decision Support Infrastructure," *ACM Sigsoft Software Engineering Notes*, 29(1):130-138, 2004, https://doi.org/10.1145/974043.974066.

[4] L. Cheung, L. Golubchik, and F. Sha, "A Study of Web Services Performance Prediction: A Client's Perspective," 2011 IEEE 19th Annual International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems, Singapore, pp. 75-84, 2011. doi: 10.1109/ MASCOTS. 2011.66.

[5] V. Datla and K. Goseva-Popstojanova, "Measurement-Based Performance Analysis of e-Commerce Applications with Web Services Components," IEEE International Conference on e-Business Engineering (ICEBE'05), Beijing, China, pp. 305-314, 2005, doi: 10.1109/ICEBE.2005.85.

[6] D Evangelin Geetha, T V Suresh Kumar, K Rajani Kanth, and Mayank P Singh, "Simulation of Multi-Tier Queuing Applications," Copyright, Diary Number: 50201/2014-CO/SW.

[7] E. Geetha D, T. S. Kumar, and K. R. Kanth, "Determining Suitable Execution Environment based on Dynamic Workload during Early Stages of Software Development Life Cycle: A Simulation Approach," *International Journal of Computational Science and Engineering*, 7(3):175, 2012, https://doi.org/10.1504/ ijcse.2012.048228.

[8] Gordon P. Gu and Dorina C. Petriu. "XSLT Transformation from UML Models to LQN Performance Models," *Proceedings of the 3rd International Workshop on Software and Performance (WOSP '02)*, Association for Computing Machinery, New York, NY, USA, pp. 227–234, 2002. https://doi.org/10.1145/584369.584402.

[9] R. Khadka, "An Evaluation of Dynamic Web Service Composition Approaches," M. J. van Sinderen, M. J. van Sinderen, B. Sapkota, and B. Sapkota (Eds.), 4th International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing - ACT4SOC 2010, SCITEPRESS, pp. 67-79, 2010.

[10] F. Lecue and N. Mehandjiev, "Towards Scalability of Quality Driven Semantic Web Service Composition," IEEE International Conference on Web Services, Los Angeles, CA, USA, 2009, pp. 469-476, 2009, doi: 10.1109/ICWS.2009.88.

[11] Y. Li, Y. Liu, L. Zhang, G. Li, B. Xie, and J. Sun, "An Exploratory Study of Web Services on the Internet," IEEE International Conference on Web Services (ICWS 2007), Salt Lake City, UT, USA, pp. 380-387, 2007, doi: 10.1109/ICWS.2007.37.

[12] Ana C. C. Machado and Carlos A. G. Ferraz, "Guidelines for Performance Evaluation of Web Services," *Proceedings of the 11th Brazilian Symposium on Multimedia and the Web (WebMedia '05)*, Association for Computing Machinery, New York, NY, USA, pp. 1-10, (2005), https://doi.org/10.1145/ 1114223.1114234,

[13] Ioana Manolescu, Marco Brambilla, Stefano Ceri, Sara Comai, and Piero Fraternali, "Model-Driven Design and Deployment of Service-Enabled Web Applications," *ACM Trans. Internet Technol.,* 5(3):439-479, August 2005. https://doi.org/10.1145/1084772.1084773.

[14] D. A. Menasce, "Scaling for e-Business," *Proceedings 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (Cat. No. PR00728)*, San Francisco, CA, USA, pp. 511-513, 2000, doi: 10.1109/MASCOT.2000. 876578.

[15] D. Menasce and V. Almeida, "Capacity Planning an Essential Tool for Managing Web Services," *IT Professional*, 4(4):33-38, July 2002, https://doi.org/10.1109/mitp.2002.1046642.

[16] R. Peña-Ortiz, A. Gil José, Julio Sahuquillo, and Ana Pont, "Analyzing Web Server Performance under Dynamic User Workloads," *Computer Communications*, 36(4):386-395, 2013, ISSN 0140-3664, https://doi.org/10.1016/j.comcom.2012.11.005.

[17] Ram Mohan Reddy, Evangelin Geetha D, Suresh Kumar T V., "Activity Based Performance Prediction for Software Systems," Technical Report, Department of MCA, M S Ramaiah Institute of Technology, Bangalore, 2016.

[18] C. U. Smith and L. G. Williams, "Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software," October 1, 2001, https://doi.org/10.1604/9780201722291.

[19] C. U. Smith, *Performance Engineering of Software Systems*, Addison-Wesley Longman Publishing Co., Inc., ISBN:978-0-201-53769-7, January 1, 1990.

[20] J. Hyun Son and M. Ho Kim, "Improving the Performance of Time-Constrained Workflow Processing," *Journal of Systems and Software*, 58(3):211-219 September 2001, https://doi.org/ 10.1016/s0164-1212(01)00039-5.

[21] Paulo S. L. Souza mail, Regina H. C. Santana, Marcos J. Santana, Ed Zaluska, Bruno S. Faical, and Julio C. Estrella, "Load Index Metrics for an Optimized Management of Web Services: A Systematic Evaluation," July 16, 2013, DOI: 10.1371 /journal. pone.0068819.

[22] N. Thio and S. Karunasekera, "Client Profiling for QoS-Based Web Service Recommendation," 12th Asia-Pacific Software Engineering Conference (APSEC'05), Taipei, Taiwan, 8 pp. 2005, Doi: 10.1109/APSEC.2005.50.

[23] P. Xiong, Z. Wang, S. Malkowski, Q. Wang, D. Jayasinghe and C. Pu, "Economical andRobust Provisioning of N-Tier Cloud Workloads: A Multi-Level Control Approach," 31st International Conference on Distributed Computing Systems, Minneapolis, MN, USA, 2011, pp. 571-580, 2011, doi:

10.1109/ICDCS.2011.88.

[24] L. D. Xu, W. Viriyasitavat, P. Ruchikachorn, and A. Martin, “Using Propositional Logic for Requirements Verification of Service Workflow,” *IEEE Transactions on Industrial Informatics*, 8(3):639-646, August 2012, https://doi.org/10.1109/tii.2012.2187908.

[25] X. Zhao, et al., “Real-Time Soft Resource Allocation in Multi-Tier Web Service Systems,” 2017 IEEE International Conference on Web Services (ICWS), Honolulu, HI, USA, pp. 492-499, 2017, doi: 10.1109/ICWS.2017.57.

Ch. Ram Mohan Reddy is currently a Professor in Department of Computer Applications at the BMS College of Engineering, Karnataka, India. He received his Ph.D. Degree in Computer Applications from Visvesvaraya Technological University, Belagavi in 2017. His current research interests include software performance engineering, data analytics machine learning. He is a member of various professional bodies.

D. Evangelin Geetha is working as Associate Professor in the Department of Master of Computer Applications. Her areas of interest are software performance engineering, predictive analytics, cloud computing, object technology and distributed systems.

R. V. Raghavendra Rao received a bachelor's and Master's from S. K. University, Anantapur, India and Madras University, India, respectively. Currently pursuing Ph.D. studies at NIT Trichy. Having 20+ years of experience in teaching, research, and development. His research articles have been published in journals and conferences related to software performance, block chain technologies, and data analytics. My research is focused on using machine learning algorithms for data analytics in medical and agricultural domains.

K. Sailaja Kumar is currently working as Senior Subject Matter Expert, Data Science, UNext, Bangalore, Karnataka, India. She received her Ph.D. Degree in Computer Applications from Visvesvaraya Technological University, Belagavi. Her areas of interest are predictive analytics, software performance, machine learning. She published a large number of publications in reputed journals.