Performance Engineering - Third chance

Performance Engineering

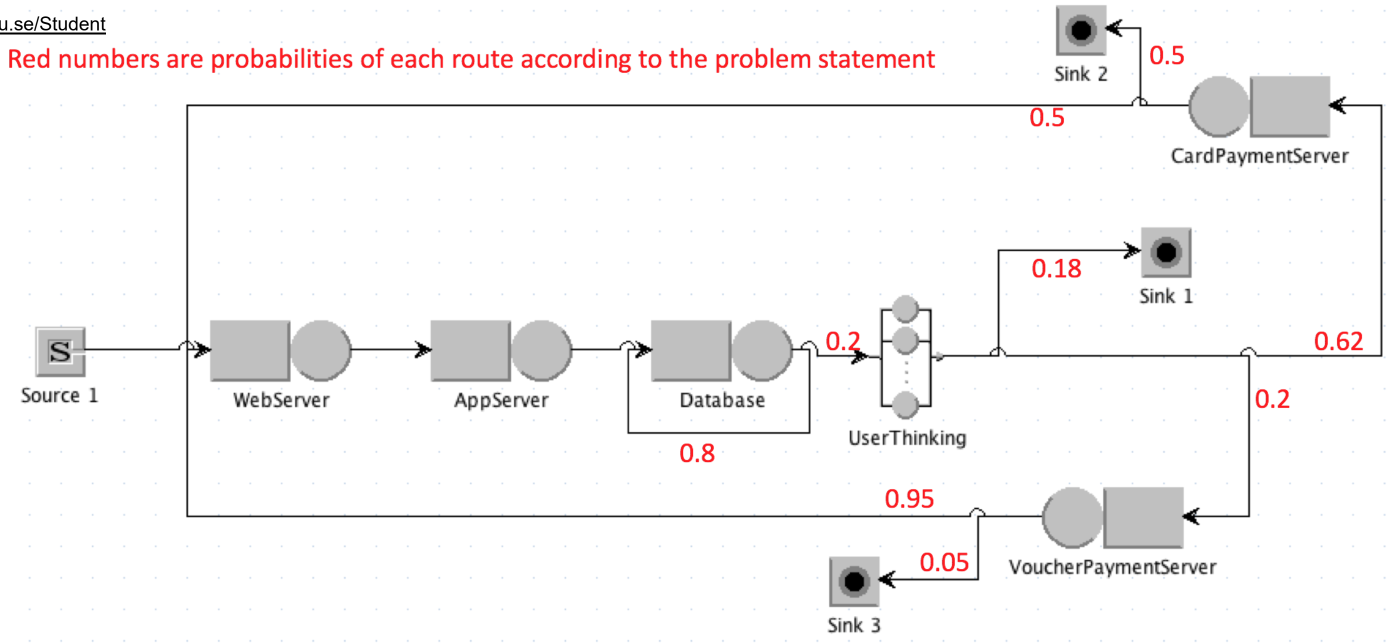
We are going to study the performance of a software system that executes web sessions from users from the Internet. The system offers an e-commerce site to the users.

We have observed the running system for 1000 minutes. During these 1000 minutes, we have seen that 72000 user sessions have been completed. Actually, we have seen that sessions arrived at an average rate of 1.2 sessions per second, and, therefore, we can deduce that all sessions that arrived were completed.

The system is composed of five service centers: A *WebServer,* an *ApplicationServer*, a *Database*, a *CardPaymentServer,* and a *VoucherPaymentServer.* There is one resource for each of the service centers. The description of the system and measured information is the following:

1. Each user's session starts executing in the *WebServer*, whose average service time is 90ms. After the *WebServer* executes, the user session continues in the *ApplicationServer.*
2. The *ApplicationServer* produces the list of articles to the user. We have observed that the utilization of the *ApplicationServer* is 52.8% and that, during a complete user session, this server receives in average 2 visits on average. To create the list of articles, each execution in the *ApplicationServer* needs to request, on average, 5 times the execution of the *Database* (it reads from the *Database* the information about articles, pricing, etc.).
3. The service time of a *Database* execution is 45ms, on average.
4. After the output of the *ApplicationServer* is produced, the user thinks for 10 seconds on average whether he/she should purchase an article. After thinking, the 62% of times the user decides to purchase an article and pay with a credit card, the 20% of times the user decides to purchase and article and pay with a Voucher, and only the 18% of times the user leaves the system without acquiring an article. The credit card payments are executed in the *CardPaymentServer,* while the payments using a voucher are executed in the *VoucherPaymentServer.*
5. The *CardPaymentServer* executes the logic for payment with credit card: receives the payment information from the user, it performs basic tests about the correctness of the data (for example, the existence of card number, no special characters in the name or address, etc.) and contacts the banking system. The average service time of this service center is 400ms.
6. The *VoucherPaymentServer* executes the logic for payment with a Voucher. We have observed that each user session generates a demand D_k in this server of 120ms. We also saw that there were, on average, 0.168 jobs in the *VoucherPaymentServer;* and that, the average time between a job arrived at the *VoucherPaymentServer* and its execution was completed (including the time that it spent waiting for service in a queue) was 0.35 seconds.
7. After the payment, the 95% of users who paid with a Voucher feel that they would like to continue purchasing items and return to the *WebServer* to start another purchase (the other 5% leave the system). However, only 50% of users who paid with actual money in the *CardPaymentServer* start another shopping (the other 50% leave the system).

The following figure shows the structure of a Queueing Network that represents the system and the routing probabilities between elements.


Red numbers are probabilities of each route according to the problem statement

You have to submit a PDF document with your answers to the following 3 exercises:

A)

A1) Use the operational laws to calculate the average service time S_k of the *ApplicationServer* and the *VoucherPaymentServer.*

A2) Assume that the population might be confined at home due to a pandemic. In that case, people will increase their online shopping because they cannot go shopping personally. Therefore, we expect that the workload to our e-commerce site will increase to an average rate of 10 user sessions per second. You can check that our current system that uses one resource in each service center is not able to deal with an average rate of 10 sessions per second. You must use the operational laws to calculate the minimum amount of resources that we need in each service center to handle an average arrival rate to the system of 10 sessions per second.

B) Model the system that has only one resource in each service center using Queueing Networks (in JMT or your preferred Queueing Network simulation engine). Simulate the model to calculate the system response time, the utilization of each service center, and the throughput of each service center. Show screenshots of the 3 types simulation results.

Hint1: In the cases that, from a service center (e.g., *WebServer*), a job can go to more than one service center, use Probabilistic Routing.

Hint2: Use the exponential distribution for all times and rates (frequencies) you need to model.

Hint3: Use the figure above, the provided structure and probabilities are correct.

C) Model the system with UML Diagrams profiled with MARTE. You are ONLY required to represent with MARTE profile the 2 next pieces of information: the workload of the system and the time that the *CardPaymentServer* needs to execute. Write the Stereotypes you need to use, the Properties you use from them, and their Values as comments attached to the corresponding elements (like the examples in the slides that we saw during the lesson). You can use your favorite tool for modeling UML Activity Diagrams. Take a screenshot of the diagram (obviously, including in the comments where you have added the performance information using MARTE)

Bonus points (10%)) In the *ApplicationServer* description, the information about "*this server receives 2 visits on average"* is redundant. You could have obtained the same value "2" using the rest of the information in the exercise. Show how the "2 visits" can be calculated from the rest of the data to get 10% extra in your assignment.

Good luck!

Status för inlämning

Försök nummer	Detta är försök 1.
Status för inlämning	Inga försök
Betygssättningsstatus	Ej betygssatt
Stoppdatum/tid	Thursday, 14 May 2020, 23:59
Återstående tid	14 timmar 13 minuter
Senast ändrad	-
Inlämningskommentarer	Kommentarer (0)

Lägg till

Du har inte gjort någon inlämning än.

◄ Performance Engineering - retake

Architecting and Design Forum ►