QNAS – a Queuing Network Animation System

Fang Pan and Jozo Dujmović
Department of Computer Science
San Francisco State University

We present design concepts, implementation, and use of a queuing network animation system (QNAS). This system is designed as a tool for the study of basic concepts of queuing networks. QNAS is a web-based numerical simulator and graphical animation system for queuing networks. It supports basic open and closed queuing network models. It is suitable as an educational tool for visualizing the dynamic behavior of queuing systems, and for testing the accuracy of analytic queuing models.

1. Introduction

Queuing models are widely used for the analysis of dynamic behavior of computer systems. Queuing phenomena are not simple and easily understandable without an insight into the nature of random events and the behavior of basic components of queuing models. These components include queues, servers, and discrete events related to random arrivals and movement of customers through service centers. Traditional performance indicators, such as response time, queue length, bottleneck resource, server utilization, throughput, interarrival time, etc. must be intuitively understood before using analytic or simulation models. In this paper we present a tool that is specifically designed to help users to visualize, understand, and model queuing phenomena. Our system is called QNAS (Queuing Network Animation System) and it is freely available through Internet [11]. In addition to animation of basic queuing models QNAS also includes a numerical simulator and can be used for testing and verification of selected analytic models.

Discrete event simulation is the only modeling method that can be used in cases where simple analytic models are not available [3, 10]. Discrete event simulators are sometimes based on specific simulation languages (e.g. GPSS [2, 6], Simscript [9], Simula [1]) or realized as specialized simulators and numerical solvers (e.g. QNAT [7], Arena [8]). Traditional discrete-event simulators are focused on numerical results and do not include animation. However, some of modern commercial discrete event simulation systems include animation as one component of their functionality (e.g. PASION [12] and Simprocess [4]). However, animation is not the primary goal of these systems because they are not designed as educational tools.

A simulator that includes a simple animation of single queue open models (gas stations, postal service, etc.) was recently developed at Tel Aviv University [13]. The animation is based on moving customers that visit several parallel servers, but the customers are not visible in the queue (the queue length is only numerically presented). Since there is no differentiation between customers it is not possible to follow the movement of a specific customer through the system.

QNAS is designed as a system that has the primary goal to show the movements of individual color-coded customers through the selected queuing network. It differs from the traditional simulators in the sense that QNAS is focused on animation and educational applications in the area of simulation of computer systems. By watching the movement of color-coded customers QNAS users can develop an intuitive feeling and understanding of queuing phenomena, which is a prerequisite for successful use of analytic models.

2. QNAS Design Concepts

In order to serve as an animation-oriented educational tool QNAS must satisfy the following design goals:

- Individual customers (jobs/tasks/processes) must be distinguished through color-coding.
- Individual customer must be visible and traceable from the moment of entering the system to the moment of leaving the system.
- Customers using queues and servers must be visible and the time they spend in queues and servers must be proportional to the actual times of the simulated real system.
- The movement of customers from resource to resource should be visible to enhance animation effects regardless the fact that the time between resources in the real system (as well as in the QNAS numerical solver) is always zero.
- The speed of moving customers must be adjustable to create various visual effects.
- User must be able to study fundamental open and closed queuing models, and have full control of their parameters (arrival rates, distributions, service times, resource numbers, etc.)

- As an educational tool, QNAS must be accessible on the Web [11].

QNAS design is based on satisfying the above goals. It includes the following basic components:

- Graphical User Interface
  - Parameter Window
  - Animation Window
  - Numerical Results Window
- Engine
  - Simulation subsystem
  - Numerical solver

QNAS supports the following operation modes:

- Animation mode
  - Single-event mode
  - Adjustable speed mode
- Maximum speed mode

The animation mode is used for visualizing queuing phenomena and for developing insight into the dynamics of discrete event systems. Users must be able to observe and develop intuitive feeling for the following fundamental performance concepts:

- Mean queue length
- Server utilization
- Throughput
- Bottleneck resource
- Saturation phenomena
- Service station residence time
- System response time
- Distribution of customers in service centers of closed models
- Discrete states of the system
- State transitions

The single-event mode is necessary in all situations where the user (either instructor or student) wants to predict the set of possible next events and then to see the event selected by QNAS. This is a valuable component of the educational process and develops a detailed understanding of operation of discrete event systems. In addition, by observing the dynamic behavior, students should be encouraged to intuitively estimate the values of performance indicators, and then to verify their prediction using the numerical simulation results.

Adjustable speed mode is necessary to show the dynamics of the processing performed by the system and to develop understanding of dynamic phenomena such as empty or saturated queues, idle servers, waiting customers, variations in service time, variations in user think time, total response time, etc.

The maximum speed mode is used to simulate large numbers of events that result in accurate numerical simulation results. These results can be used to verify the results predicted by analytic models, and to evaluate the accuracy of results that can be expected from discrete-event simulators. In the maximum speed mode, the animation is suspended and the simulator generates new events and numerical results at the maximum speed supported by the available computer.

In order to provide Web access, and to integrate the GUI design with a numerical simulator, QNAS should be implemented in Java. The analyzed queuing models can be either arbitrary (based on drag and drop composition of queuing models), or based on a finite set of carefully selected models. A universal model composer greatly increases the complexity and the size of the system, as well as the model set-up time, without substantially increasing the system utility. Consequently, QNAS is designed as an expandable simulator that currently includes the following basic queuing models:

- Single server model (Fig. 1a).
- Single server with feedback (round-robin, Fig. 1b)
- Two servers model (Fig. 1c).
- Machine repairman (interactive system: multiple workstations and multiple processors, Fig. 1d).
- Central server model with single processor, disk controller, and multiple disks (Fig. 1e).

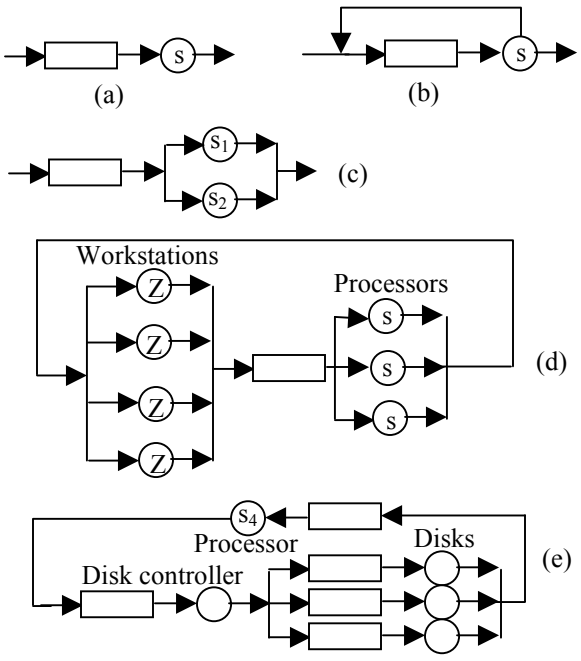


Figure 1. Basic open and closed queuing models

3. QNAS Implementation

QNAS is implemented as a Java applet. It can be installed either locally or on a remote web server. Used as an Internet educational tool QNAS is typically installed on a web-server, and the user does not need to install any program. At the run time, the user downloads the QNAS applet across the Internet and executes it locally in the browser. The longest applet load time (for 33 kbps modems) is 45 seconds. After loading, the QNAS applet is running completely within the client machine browser sandbox. So, there is no further network traffic needed between the server and the client machines. As the result, there is no animation distortion after the applet starts running.

Functional components of QNAS are shown in Fig. 2. The core component of QNAS is a traditional discrete event simulator. It is closely connected with GUI subsystem, animation subsystem and a numerical solver. The numerical solver is used to compute numerical values of all performance indicators (server utilization, response time, throughput, queue length, etc.) Numerical results are passed to the GUI subsystem and displayed in a special result window.

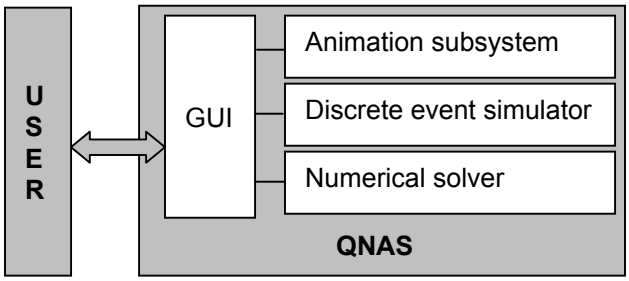
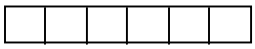


Figure 2. User interaction with QNAS

From the educational standpoint the most important part of QNAS is the animation subsystem. This subsystem implements the following eight graphical components:

- **Job** (displayed as a color-coded circle):

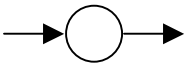
- **Empty queue** (each box can contain a job):



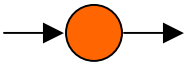
- **FCFS queue with jobs**:

- **Connection line** showing the direction of movement of jobs:

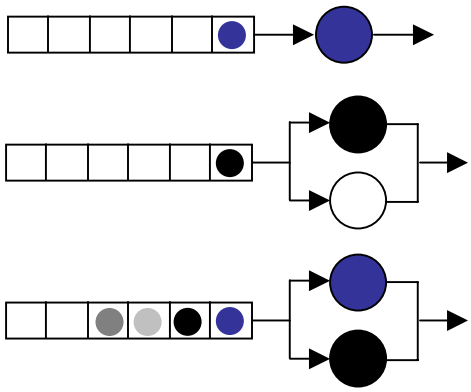
- **Idle server** (empty circle):



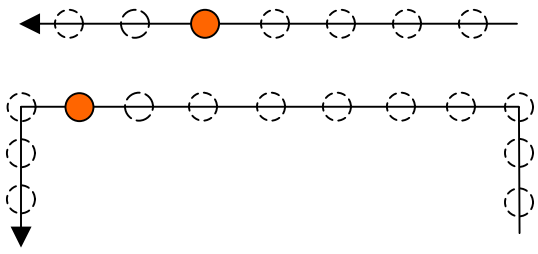
- **Busy server** (the color of the circle identifies the job):



- **Job being served** by a server is represented as a queue containing a job at the head of queue position and a server containing the same color code as the job at the head of queue. In the case of multiple servers either all servers are busy or some are busy and some are idle:



- **Job path** is modeled as a line with a sequence of hidden job positions. At any time only one position is color-coded with the color of a moving job. The color-coded job is moving across the line at the constant speed that is selected for optimum visual effect. Job paths can consist of one or more linear segments, as follows:



QNAS animation strictly follows the movement of each job through the network of queues and servers. When the job leaves one network component and moves towards another component using a job path, this must be visible and must take sufficient time so that the user can follow the movement. There is an optimum speed of this movement: if the movement is too fast the animation effect is lost, and if the movement is to slow then the slow motion animation is boring and user's attention is quickly lost. QNAS automatically initializes the animation speed and provides user's control to dynamically adjust the optimum speed.

A typical state transition process is illustrated in Fig.3. The state before transition consists of two jobs in service: the black job in the upper subsystem and the gray job in the lower subsystem. Let the next event be the end of service for the black job in the upper subsystem. In this case the black job will leave the upper subsystem and will move to the tail of the lower subsystem queue. This transition is animated as shown in Fig. 3: we first clear the color of the upper subsystem server and then the terminated job quickly moves through the sequence of adjacent locations on the path that connects the upper and the lower subsystems. At the end of transition the black job is stored as the tail job of the lower FCFS queue. Since the lower subsystem has two servers the idle server is allocated to the black job and the service of the black job in the lower subsystem starts immediately upon its arrival. The presented transition has a sequence of transition states that are interpolated between the initial state (before transition) and the final state (after transition). Obviously, the movement of the black job along the job path must take the time necessary to generate a desired visual effect of fast movement.

It is important to note that the numerical solver must neglect the animation effects of job path because in queuing models the movement of jobs between service stations (frequently equivalent to context switching) is always neglected. Consequently, QNAS generates both pleasant dynamic visual effects and correct numerical results.

3. Five Queuing Models Supported by QNAS

QNAS supports five basic open and closed queuing network models: the single server (Fig. 4), two servers (Fig. 5), round robin (Fig. 6), machine repairman (Fig. 7), and the central server (Fig. 8). Interarrival and service times can have exponential distribution, or uniform distribution, or a constant value. QNAS also generates numerical simulation results that can be used to verify analytic models.

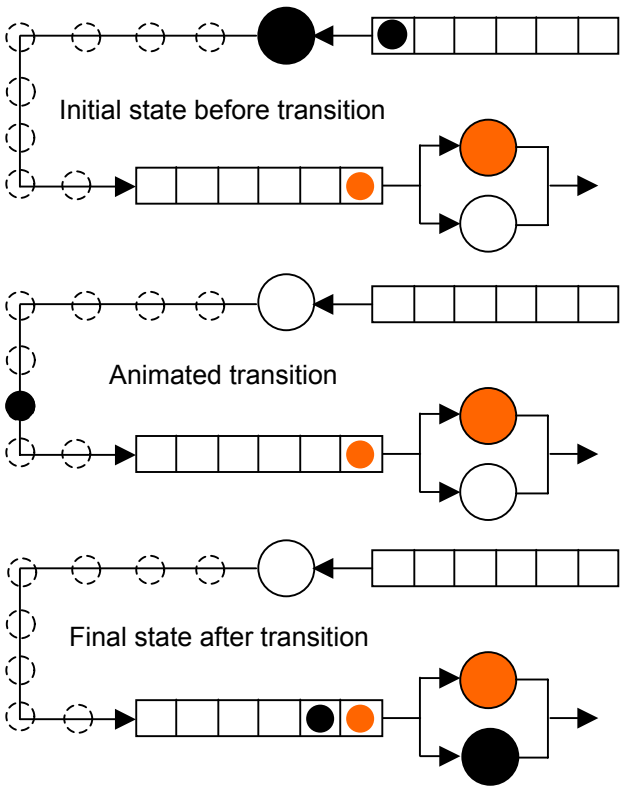


Figure 3. Typical animation of a state transition process (an initial state, transition, and the final state)

All QNAS models include queues that are internally sized to store a large number of jobs (e.g. 500 or more). However, the graphical presentation of queues during animation is limited. For example, a typical QNAS window showing a central server model in execution inside a browser is shown in Fig. 8. This model shows a central processor, a disk controller, and 8 disk units; some disks are the bottleneck devices and have long queues of waiting jobs. The length of all displayed queues in this model is limited to 12 jobs, and if the number of jobs in the queue is greater than 12 then only the first 12 jobs are displayed on the screen. Similar concept holds for all other models.

The first step in using a selected model is to initialize its parameters. A click on the Initialize button opens the setup window. This is exemplified in Fig. 4 for the case of single server model that has interarrival times uniformly distributed between 0 and 10 seconds and service times uniformly distributed between 1 and 5 seconds. At the end of parameter definition the user must click the Accept button in order to save the parameters of the model. After this the Setup window closes and the user can run the selected model. Fig. 4 also shows various control buttons, such as Help.

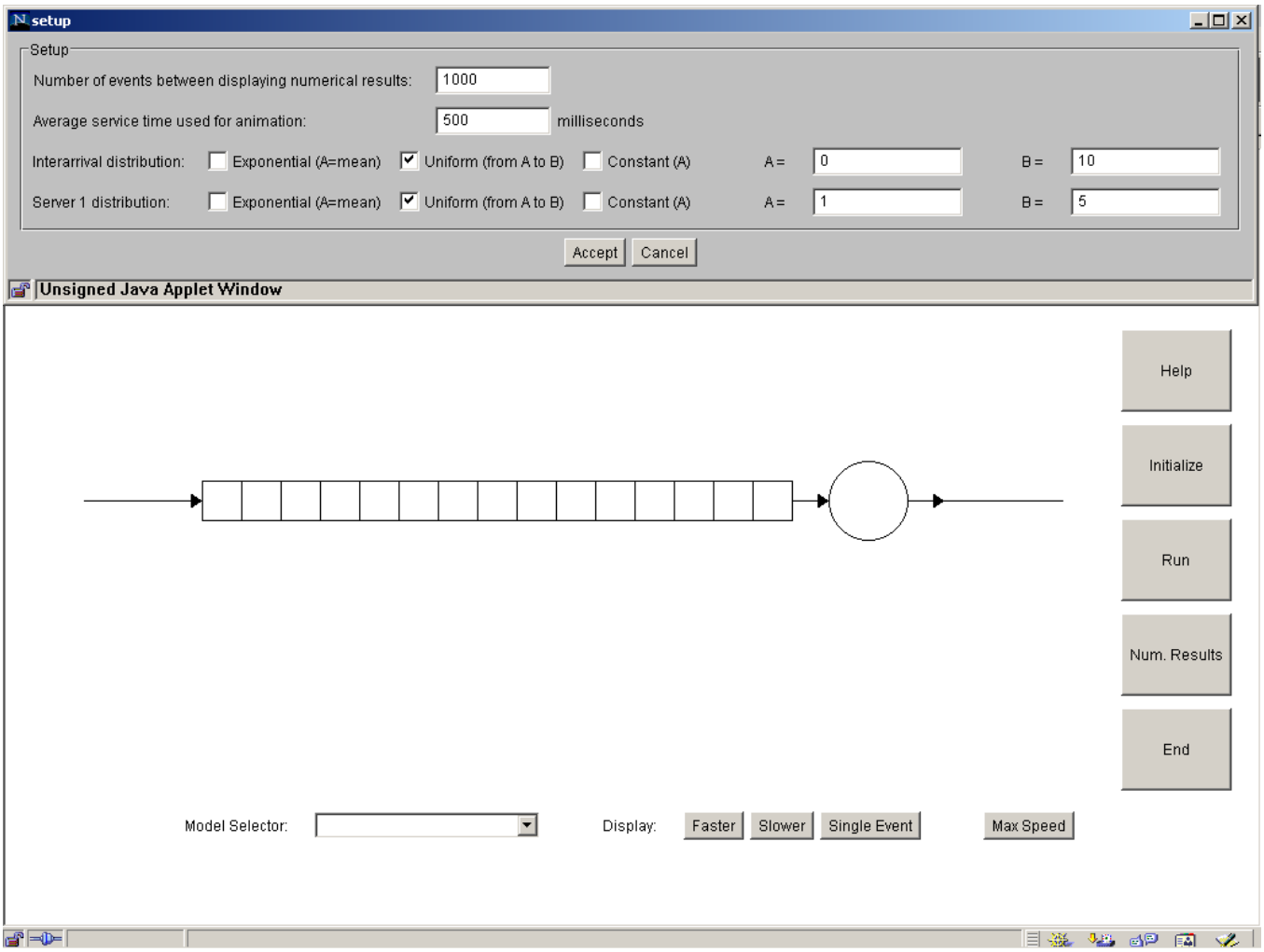


Figure 4. A single server model and its parameter initialization window

QNAS runs in two fundamental execution modes: (1) Animation mode (default) and (2) Max Speed mode. The Animation mode is used to visually present queuing phenomena with color-coded jobs that can wait in queues, or use servers, or move between service centers. The animation process has a dynamically adjustable speed that also includes the Single Event speed. The speed of animation does not affect numerical results. The Max Speed mode is used to suspend animation and generate numerical results at the highest speed. The operation of QNAS is controlled by buttons that have the following roles:

Initialize

This button is used to activate the setup window (Fig. 4). This window us used for definition of all parameters of the selected model. These parameters include the topology of the model (number of servers) and distributions of interarrival and service times. The distributions can be exponential, uniform, or (in a deterministic cases) all values can be constant.

Exponential distribution has only one parameter: the average value *A*. Uniform distribution generates values between the minimum value *A* and the maximum value *B*. A constant interarrival or service time is denoted *A*. For example, to simulate the M/D/1 model, we select an exponentially distributed interarrival time, and a constant service time. The parameters *A* and *B* do not assume any specific time units. The user can assume any type of time units. The assumed units will be used when calculating and displaying numerical results.

Initialization can be done either at the beginning of session (Fig. 4) or after pressing the *End* button. Initialization is disabled during other QNAS states (only the shadow of the Initialize label is visible, as shown in Fig. 5).

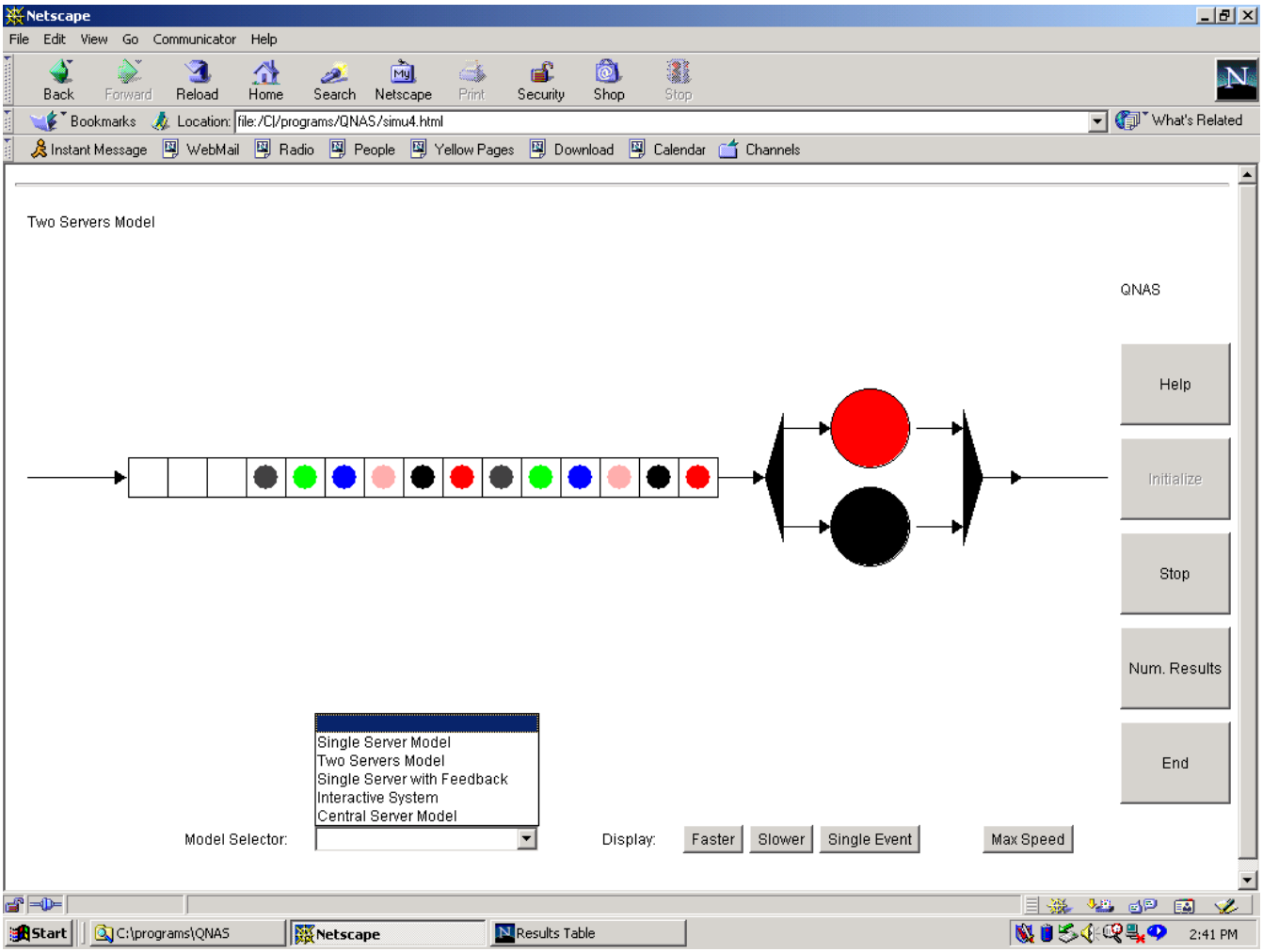


Figure 5. A model with two parallel servers (12 jobs in the FCFS queue, 2 jobs in service)

The simulator generates numerical results. The number of events between updating numerical results is an adjustable parameter. To adjust the speed of animated simulation, select the average animation service time (you can use the default value and then make animation faster or slower during the execution).

Run/Stop toggle button

The *Run* button is used to start state transitions after initialization or a pause caused by *Stop* button. After pressing *Run* the label of the button becomes Stop. When the simulator is running *Stop* is used to make a pause, and display/analyze the current state.

Numerical Results

The *Num. Results* button is used to display the current values of the following parameters:

JOBS = the number of jobs that have been served.
TIME = the total simulated time (total time from the beginning of simulation) expressed in the same units as distribution parameters.
ATA = average inter-arrival time.
SDTA = standard deviation of inter-arrival time.
ATS = average service time.
SDTS = standard deviation of service time.
R, ART = average response time.
SDRT = standard deviation of the response time.
AJOB = average queue length.
SDJOB = standard deviation of the queue length.
U = sever utilization (from 0 to 1).
X = throughput (transactions per time unit).

End

The *End* button is used to terminate simulation. After pressing *End* it is possible to initialize QNAS with new simulation parameters.

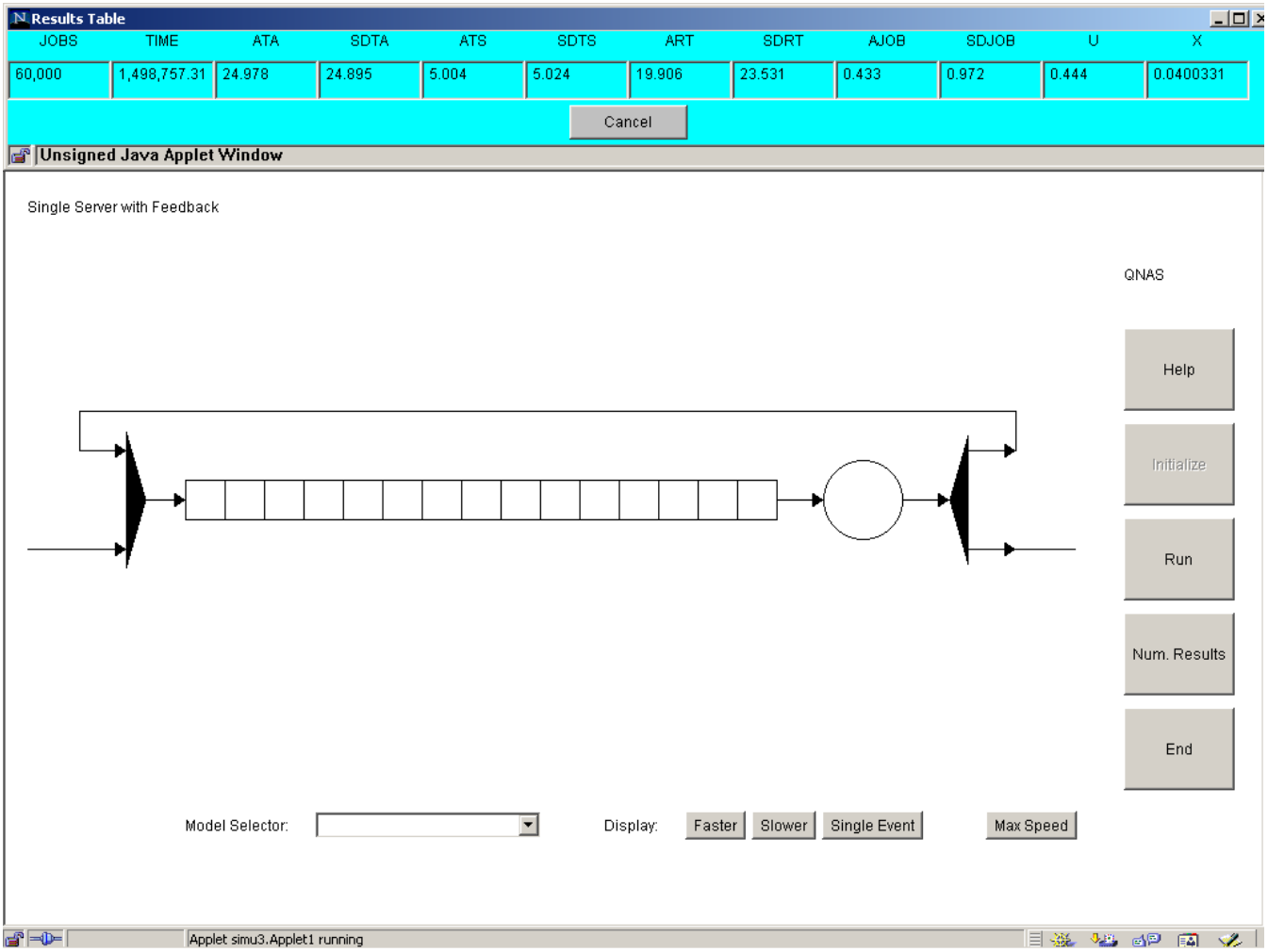


Figure 6. A single server model with feedback and its numerical results window.

Faster button

The *Faster* button is used to increase the speed of animation, or to transfer from the *Single Event* mode to the standard animation mode. The speed of animation does not affect numerical results. *Stop* and *End* are also valid options in this mode.

Slower button

The *Slower* button is used to decrease the speed of animation, or to transfer from Max Speed Mode to the standard animation mode. The speed of animation does not affect numerical results. Stop and End are also valid options in this mode.

Single Event

The *Single Event* is used to display a next event in the simulated queuing system. The event can be either the end of serving a job by a server (job departure), or the arrival of a job in a queue. In some cases, the departure from a server and the arrival in a queue can be the same event. After receiving the Single Event request, QNAS displays the single transition from a current state to a new state, and waits for the next *Single Event* request or transition to another simulation mode (*Faster* or *Max Speed*). *Stop* and *End* are valid options that can be used in this mode.

Max Speed

The *Max Speed* button is used to suspend animation and generate numerical results at the maximum speed the available computer can support. In the *Max Speed* mode the possible transitions include *Slower* (return to the standard animation) or the *Single Step* mode. *Stop* and *End* are also valid options.

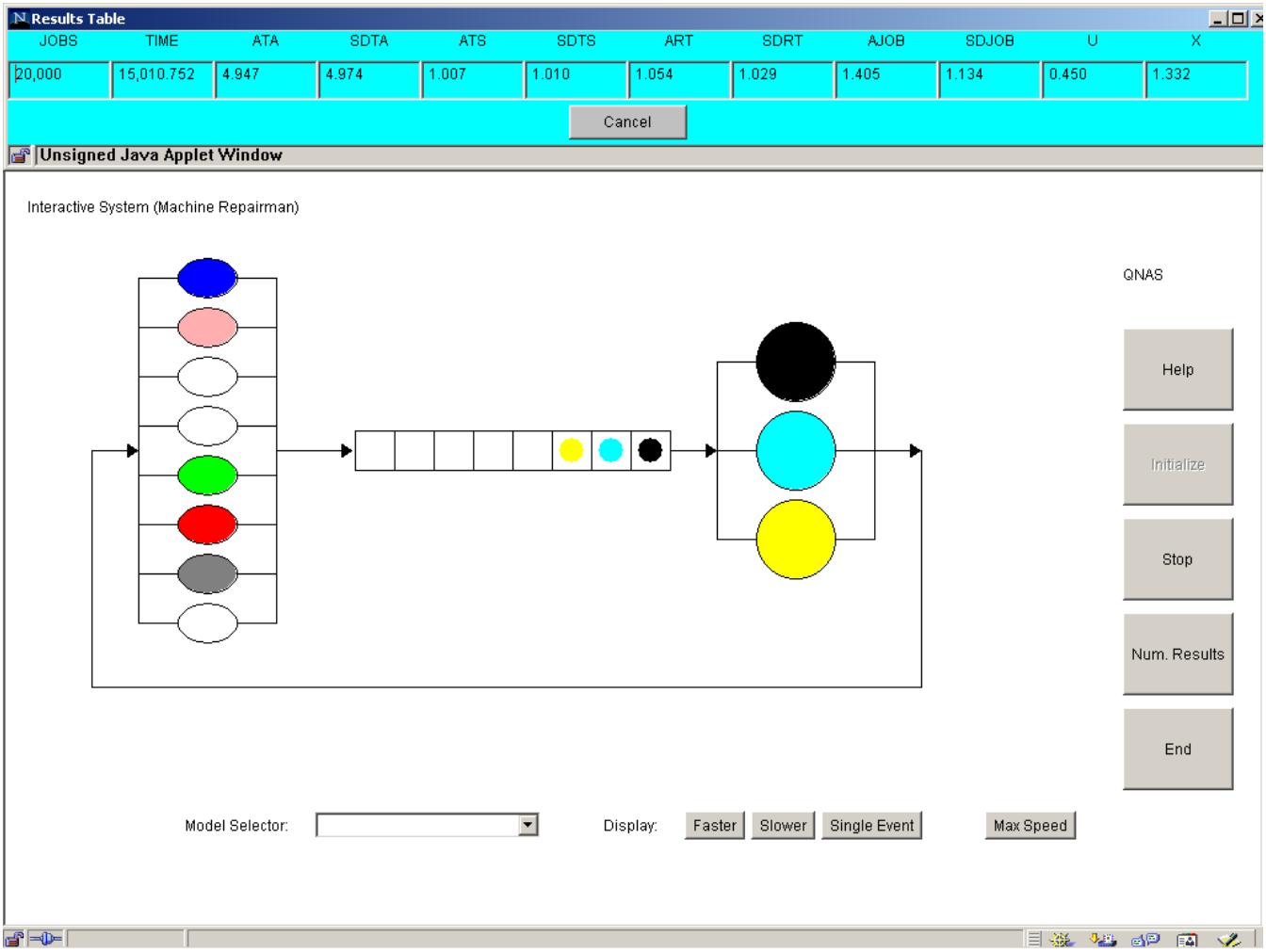


Figure 7. An interactive system with eight workstations and three processors

Model Selector button

Model Selector is a selector widget that provides for selection of queuing models supported by QNAS.

4. Numerical Experiments

Numerical problems that can be addressed using QNAS include: (1) estimation of performance indicators for supported queuing models, (2) verification of approximate queuing models, and (3) the study of the accuracy of simulation models (in cases where exact analytic results are available).

Experiment 1. To illustrate the numeric capabilities of QNAS let us first analyze a single server system used as a model of processor that receives identical data packets that form a Poisson arrival process with the average arrival rate X=10 sec^{-1}. If the processing time per packet is uniformly distributed from 0.01 to 0.05 sec, than the average service time is S=0.03 sec, its coefficient of variation is $v_s = 2/\sqrt{27}$ and this system can be modeled using the M/G/1 model. The processor utilization is U=SX=0.3 and the response time is given by the Pollaczek-Khintchine formula:

$$R = \frac{S}{1-U}\left[1 - \frac{U(1-v_s^2)}{2}\right] = 0.0374 \text{ sec}$$

The results generated by QNAS after 30000 jobs are U=0.299 and R=0.0375 sec. If the processing time is constant, S=0.03 sec, then $v_s = 0$ and we have the M/D/1 model. The response time is R=S(1-U/2)/(1-U) = 0.0364 sec. QNAS result is R=0.0364 sec.

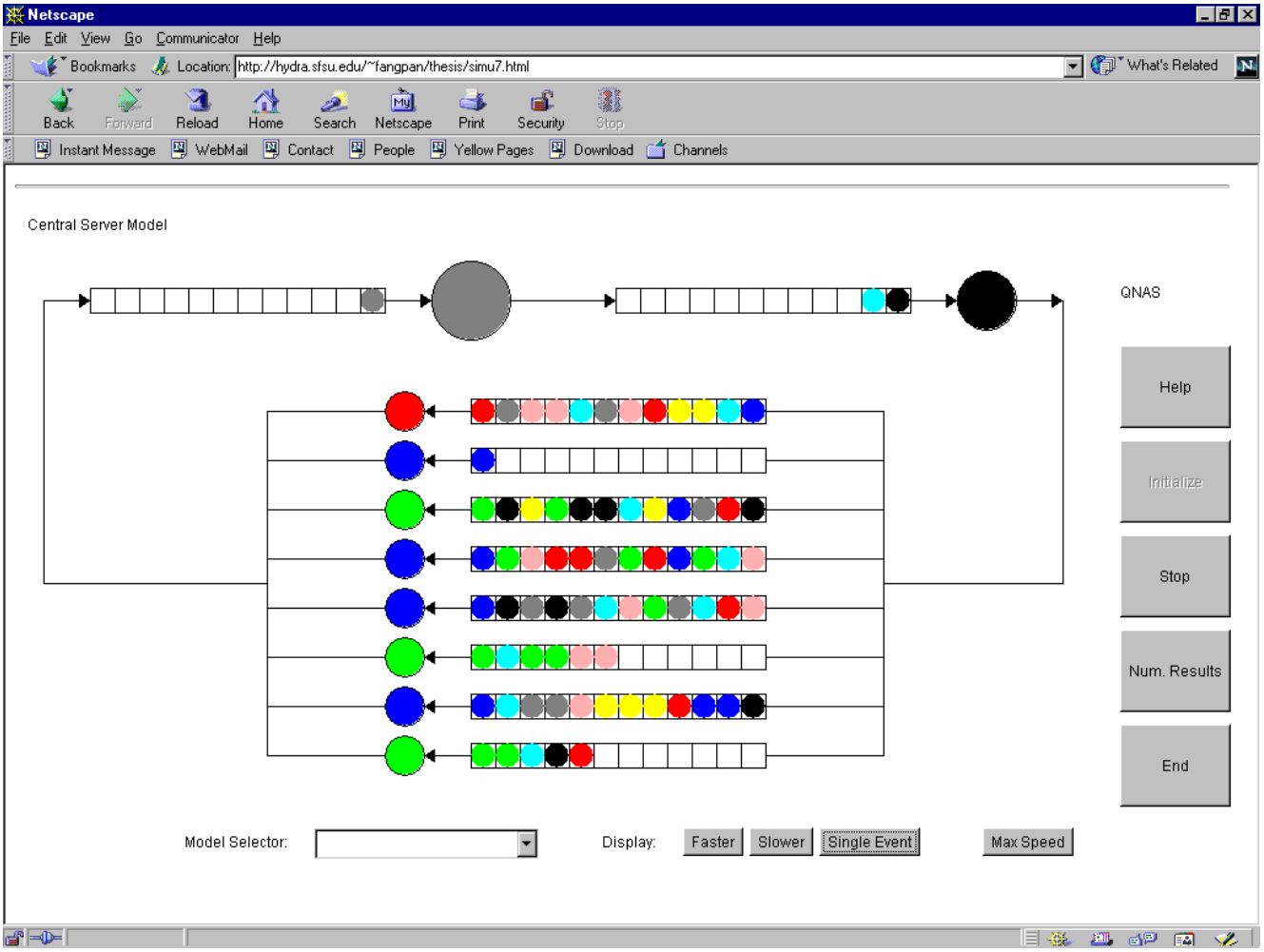


Figure 8. A central server model with eight disk units (the case of disk bottlenecks)

Experiment 2. Our second example illustrates the use of QNAS for the verification of an analytic model. We again consider a single server system. If the interarrival time *a* is uniformly distributed from 1 to 10 seconds and the service time is uniformly distributed from 1 to 5 seconds, than this system can be modeled using a G/G/1 model. We have the mean service time S=3 sec, throughput X=0.2 sec^{-1}, processor utilization U=SX=0.6, and the coefficients of variation of interarrival and service times $v_a = 1/\sqrt{3}$ and $v_s = 2/\sqrt{27}$. Our approximate G/G/1 response time formula is:

$$R = \frac{S}{1-U}\left\{1-\frac{U}{2}\left[1-v_s^2-\frac{(v_s^2+1)(v_a^2-1)}{U^2v_s^2+1}\right]\right\} = 3.95 \text{ sec}$$

QNAS simulation results are U=0.6 and R=4.07 sec indicating that our approximate formula in this case predicts the response time with an error of 3%.

Experiment 3. The single server feedback model represents open queuing networks with feedback. If the recycling probability is *p* then the response time of the exponential version of this model is R=S/(1-p-SX) and the server utilization is U=SX/(1-p). For example, if p=0.55, S=0.005 sec, and X=40 sec^{-1} then U=0.4444 and R=0.02 sec. QNAS results after 60000 jobs are U=0.444 and R=0.0199 sec. Let us now investigate the same system in the case where the service time is uniformly distributed between 0 and 0.01 second; the numeric results are U=0.445 and R=0.0187 sec, showing the reduction of response time. If the interarrival time is constant, 1/X=0.025 sec then the corresponding results are U=0.441, R=0.0121 sec. This indicates that the variability of random variables is responsible for queuing effects and the growth of the response time. To confirm this conclusion we now eliminate the variability of service time using the constant value S=0.005 sec. The corresponding numeric results are U=0.444, R=0.0118 sec. These

examples can also be used for evaluating the robustness of exponential queuing models.

Experiment 4. An interactive system (n work-stations, m processors, an average think time Z, and an average processor time S) can be analyzed using a classical exponential model. The model yields the following formulas for processor utilization U, throughput X, and response time R:

$$R = \frac{ns}{mU} - Z, \quad X = \frac{n}{R+Z}, \quad U = 1 - \sum_{i=0}^{m-1} \frac{m-i}{m} p_i$$

$$p_0 = \frac{1}{1 + \sum_{k=1}^{n} \prod_{j=1}^{k} \frac{n+1-j}{\min(m,j)} \left(\frac{S}{Z}\right)^k}$$

$$p_k = p_0 \prod_{j=1}^{k} \frac{n+1-j}{\min(m,j)} \left(\frac{S}{Z}\right)^k, \quad k > 0$$

In the case where n=8, Z=5 sec, S=1 sec, and m=1, the results of the above exponential model are U=0.93, R=3.603 sec and X=0.93 sec^{-1}. QNAS results after 20000 jobs are almost the same: U=0.93, R=3.623 sec and X=0.93 sec^{-1}. In the case of m=2 processors we have R=1.314 sec and for m=3, R=1.045 sec; the equivalent QNAS results are R=1.312 and R=1.054 sec respectively. In the case of one processor and constant processor time S=1 sec, QNAS reports R=3.24 sec, i.e. a faster response than in the case of random service time.

Experiment 5. The central server model shown in Fig. 1e consists of K=5 resources. The service times for disks are $S_1 = 6$ msec, $S_2 = 8$ msec, and $S_3 = 10$ msec. The processor service time $S_4 = 2$ msec, and the disk controller service time is $S_5 = 3$ msec. The numbers of visits to CPU and disks per job are $V_4 = 201$, $V_1 = 90$, $V_2 = 60$, $V_3 = 50$. The corresponding total disk demands per job are $D_1 = V_1 S_1 = 0.54$ sec, $D_2 = V_2 S_2 = 0.48$ sec, $D_3 = V_3 S_3 = 0.5$ sec. The numbers of visits to processor and disk controller are $V_4 - 1 = V_5 = V_1 + V_2 + V_3 = 200$ yielding the demand for CPU $D_4 = V_4 S_4 = 0.402$ sec and the demand for controller $D_5 = V_5 S_5 = 0.6$ sec per job. If the degree of multiprogramming is N=10 then we can analyze this system using the classic load independent Mean Value Analysis model shown in Fig. 9. The parameters of this model are resource queue lengths Q_1, Q_2, Q_3, Q_4, Q_5, response times R_1, R_2, R_3, R_4, R_5, throughputs X_1, X_2, X_3, X_4, X_5, and utilizations U_1, U_2, U_3, U_4, U_5. The resulting system throughput is X [jobs/sec] and each job response time is R [sec].

$Q_k(0) = 0, \quad k = 1,...,K$

for n=1 **to** N
{

$R_k(n) = S_k[1 + Q_k(n-1)], \quad k = 1,...,K$

$R(n) = \sum_{k=1}^{K} V_k R_k(n)$

$X(n) = n / R(n)$

$X_k(n) = V_k X(n), \quad k = 1,...,K$

$U_k(n) = S_k X_k(n) = D_k X(n), \quad k = 1,...,K$

$Q_k(n) = V_k X(n) R_k(n), \quad k = 1,...,K$

}

Fig. 9. A traditional load independent MVA model

The MVA model for N=10 yields the following results:

$U_1 = 74.7\%, \quad U_2 = 66.4\%, \quad U_3 = 69.1\%,$
$U_4 = 55.3\%, \quad U_5 = 83\%$
$Q_1 = 2.263, \quad Q_2 = 1.689, \quad Q_3 = 1.861,$
$Q_4 = 1.142, \quad Q_5 = 3.045$
$R_1 = 0.0182\,\text{sec}, \quad R_2 = 0.02035\,\text{sec}, \quad R_3 = 0.0269\,\text{sec},$
$R_4 = 0.00413\,\text{sec}, \quad R_5 = 0.01101\,\text{sec}$
$X = 1.383\ job/\text{sec}$
$R = 7.231$ sec

The critical degree of multiprogramming is

$$N^* = \frac{(D_1 + D_2 + D_3 + D_4 + D_5)}{\max(D_1, D_2, D_3, D_4, D_5)} = 4.2 \text{ jobs.}$$

The disk controller has the largest queue length and utilization; consequently, it is the bottleneck device.

Figure 10 shows the initialization screen for the central server model with all parameters. After 2505 jobs QNAS reports the results shown in Fig. 11. The errors of simulation results compared to analytic results are typically 1% or less.

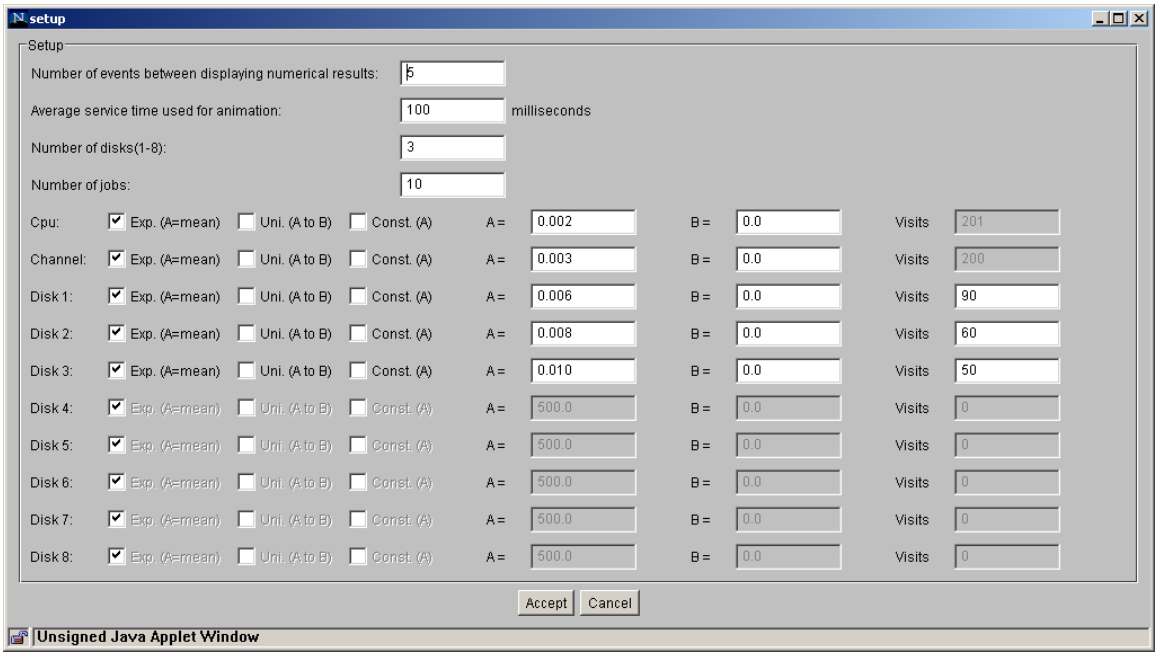


Figure 10. Definition of parameters for a sample central server model

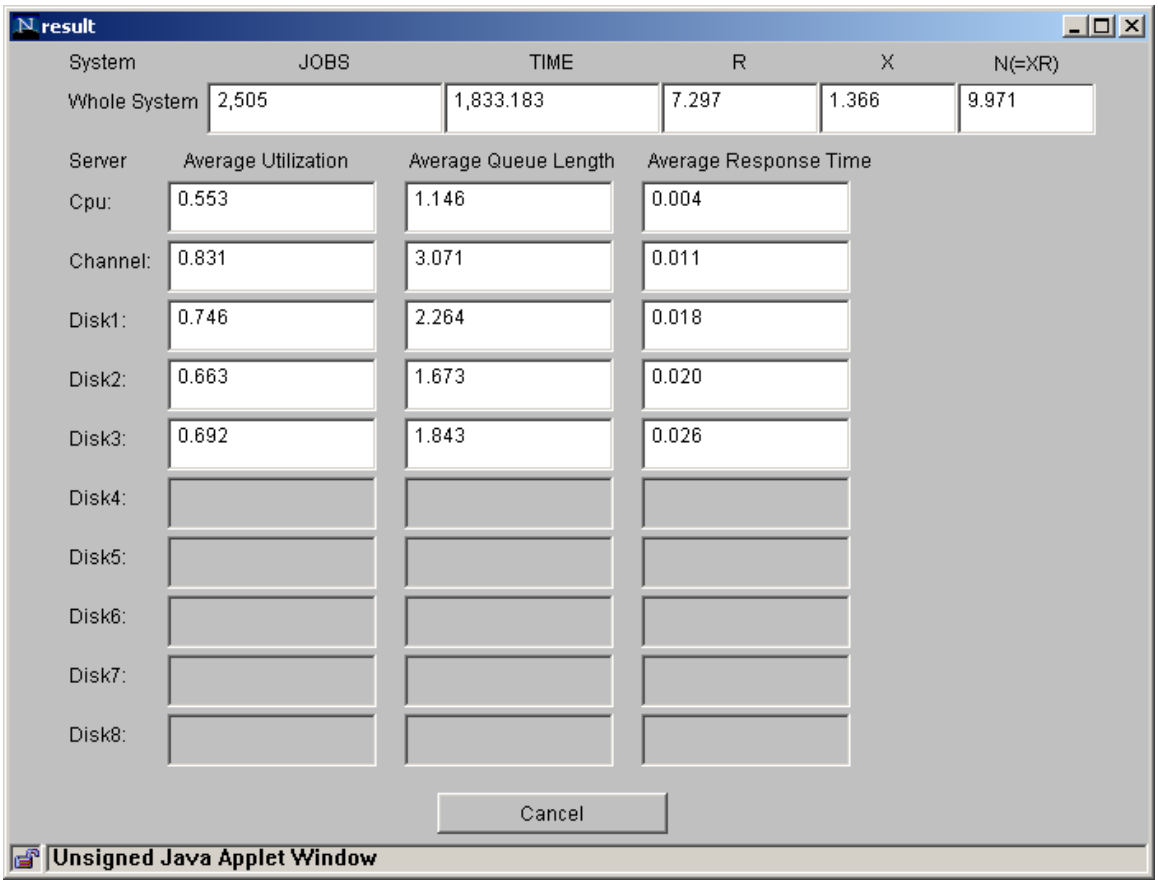


Figure 11. Numerical results of the central server model after 2505 jobs

5. Conclusions and Future Work

In this project Java was proved to be an appropriate environment for the development of simulation and animation systems. Graphical components of QNAS can be efficiently implemented in Java using either AWT or Swing. To reduce the download time when QNAS is used as an applet the current version of QNAS is based on AWT. The download time of the complete QNAS is from 4 seconds over DSL, to 45 seconds over a 33 kbps modem connection. Of course, Java also provides a full cross-platform portability.

QNAS can be efficiently used in the following areas:

- Qualitative analysis and visual demonstration of queuing phenomena.
- Development of intuitive feeling for all major performance indicators of queuing systems.
- Evaluation of approximate queuing formulas.
- Investigation of the robustness of selected analytic models.

The first two items in the above list have educational value and have been successfully tested in the classroom. The last two items indicate a possible research application of QNAS.

Service time distributions can be independently adjusted for all parallel servers. QNAS offers three cases of interarrival and service times: an adjustable range uniform distribution (denoted U), an adjustable exponential distribution (denoted M), and a constant value (denoted D). In the case of single server models this gives 9 combinations: M/M/1, M/D/1, M/U/1, D/M/1, D/D/1, D/U/1, U/M/1, U/D/1, U/U/1. All of them are special cases of the G/G/1 model, and the U/U/1 case can be used for testing the validity of approximate G/G/1 formulas. Similarly, there are 9 possible combination for the round robin model, 27 combinations for the two-server model, 9 combinations for the machine repairman model, and 59049 combinations for the 8-disk central server model. Consequently, QNAS offers 59103 possible combinations of distributions. However, user-defined arbitrary distributions, as well as user-defined queuing network configurations, are not yet implemented.

In the future QNAS can be expanded in several ways. First, more models can be included, particularly in the area of load-dependent servers and non-FCFS queues that are necessary for modeling disk units [5]. Second expansion can be in the area of providing API so that users can plug-in user-defined distribution models. Finally, the existing queuing models can be expanded to include more details and system parameters, as well as the capability to simulate arbitrary user-defined configurations of queuing networks.

References

[1] G.M. Birtwistle. *Discrete Event Modelling on Simula*, Caci Products Co., 1979.

[2] P.A. Bobillier. *Simulation With GPSS and GPSS V* , Prentice-Hall, 1976.

[3] Paul Bratley. *A Guide to Simulation*, Springer Verlag, 1987.

[4] CACI, *Simprocess* http://www.simprocess.com/simprocess.cfm

[5] Dujmović, J.J, D. Tomasevich, and M. Au-Yeung, *Measurement and Modeling of Disk Subsystem Performance*. The Computer Engineering Handbook, edited by V. Oklobdzija, CRC Press, pp. 8-1 – 8-20, 2002.

[6] Gordon, G. *The Application of GPSS V to Discrete System Simulation*, Prentice-Hall, 1975.

[7] Kaur, H.T. D. Manjunath, and S,K. Bose, *The Queuing Network Analysis Tool (QNAT).* Proceedings of MASCOTS 2000, pp. 341-347.

[8] David Kelton, *Simulation With Arena*, McGraw Hill, 1997.

[9] Law, A., *Introduction to Simulation Using Simscript Ii.5*, CACI Products Co., 1984.

[10] I. Mitrani. *Simulation Techniques for Discrete Event Systems* (Cambridge Computer Science Texts, 14)", University Press, Cambridge, 1982.

[11] QNAS: http://violin.sfsu.edu/QNAS .

[12] Raczynski, S., *PASION Simulation System*. http://www.raczynski.com/pn/pn.htm

[13] Shalgi, R, I. Guedj, and G. Leonov, *Queue Simulation*. http://www.math.tau.ac.il/~rshalgi/final/