System Execution Models

Outline

- System execution model basics
- Performance metrics
- Solving the system execution model
- Advanced system execution model
- Case study – modeling distributed systems

Introduction

- Software execution model
  - Provides a static analysis of the mean, best- and worst-case response times for software
  - Characterizes the resource requirements of the proposed software alone, in the absence of other workloads or multiple users
- If the predicted performance in the absence of these additional performance-determining factors is unsatisfactory, then there is no need to construct more sophisticated models

System Execution Models

- The system execution model characterizes the software’s performance in the presence of dynamic factors, such as other work loads or multiple users
- The system execution model aims to solve the **contention for resources**
- If the software execution model indicates that there are no problems, then you are ready to construct and solve the system execution model to account for contention efforts

Sources of Contention for Resources

- Multiple users of an application or transaction executing at one time, e.g. several ATM customers do a withdrawal simultaneously
- Multiple applications or systems executing on the same hardware resources at one time
- The application under consideration can have separate concurrent processes
- The application may be multi-threaded to handle concurrent requests for different external processes

Benefits of System Execution Model

- Elementary system execution models provide
  - More precise metrics that account for resource contention
  - Sensitivity of performance metrics to variations in workload composition
  - Scalability of the hardware and software to meet future demands
  - Effect of new software on service level objectives of other systems
  - Identification of bottleneck resources
  - …

System Model Basics

- Represents the key computer system resources as queues and servers
  - A **server** represents a component of the environment that provides some service to the software
  - A **queue** represents jobs waiting for service

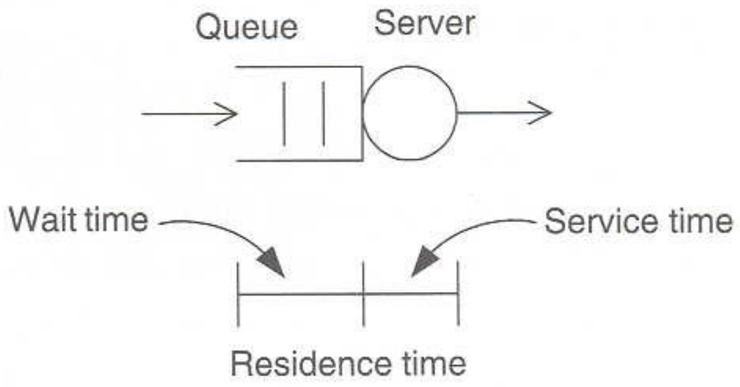


Figure 6-1: Queue-Server Representation of a Single Computer System Resource

Performance Metrics

- Performance metrics of interest for each server are
  - ***Residence time***, RT: the average time jobs spend in the server, in service and waiting
  - ***Utilization***, U: the average percentage of the time the server is busy
  - ***Throughput***, X: the average rate at which jobs complete service
  - ***Queue length***, N: the average numbers of jobs at the server (receiving service and waiting)

Example 6-1: Metric Data

- From the execution profile in Figure 6-2, we obtain the following data:

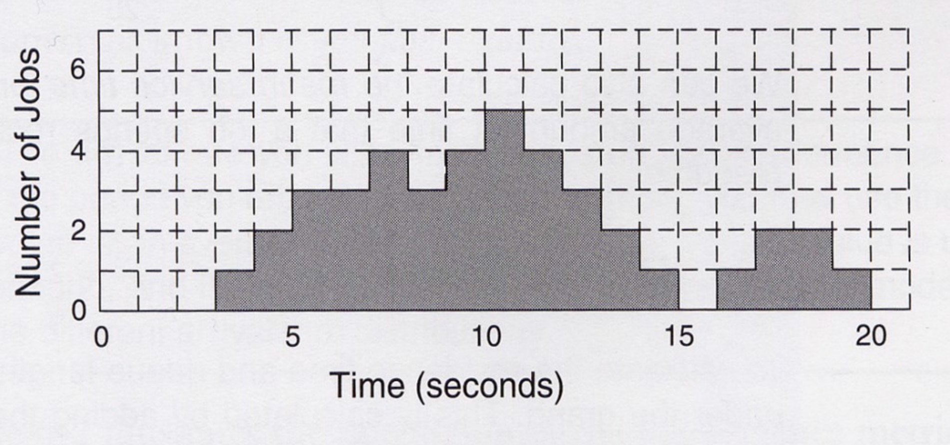


Figure 6-2: Hypothetical Execution Profile

Metric	Value
Measurement period (T)	20 sec
Number of arrivals (A)	8 jobs
Number of completions (C)	8 jobs
Busy time (B)	16 sec

Example: Calculation of Performance Metrics

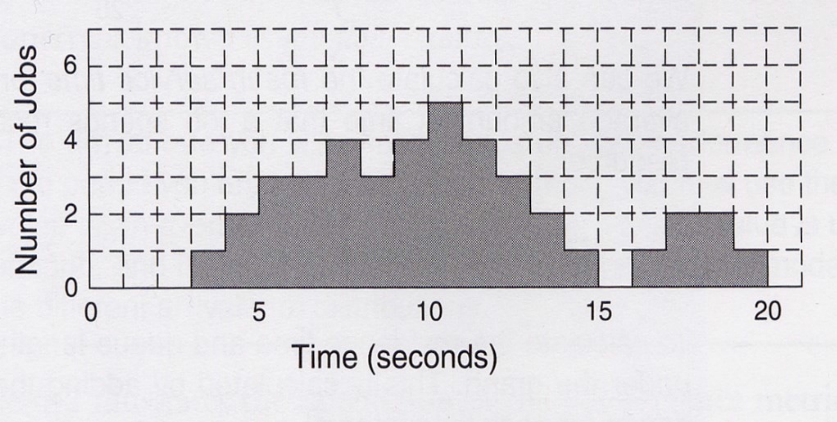


Figure 6-2: Hypothetical Execution Profile

Metric	Value
Measurement period (T)	20 sec
Number of arrivals (A)	8 jobs
Number of completions (C)	8 jobs
Busy time (B)	16 sec

- Utilization: $U = B/T$ — $U = B/T = 16/20 = 0.8$ jobs/sec
- Throughput: $X = C/T$ — $X = C/T = 8/20 = 0.4$ jobs/sec
- Mean service time: $S = B/C$ — $S = B/C = 16/8 = 2$ sec
- Area Under graph: $W = \sum_{\text{time}}(\#\text{jobs})$ — $W = \sum_{\text{time}}(\#\text{jobs}) = 41$ jobs
- Residence time: $RT = W/C$ — $RT = W/C = 41/8 = 5.125$ sec
- Queue length: $N = W/T$ — $N = W/T = 41/20 = 2.05$ jobs

Note: Queue length N includes the job under processing as well waiting.

Solving the Queueing Model

- What if no data on a new development project?
  - Use similar calculations, based on predicted ***workload intensity*** and ***service requirements***
- ***Workload intensity***
  - A measure of the number of requests made by a workload in a given time interval
- ***Service requirements***
  - The amount of time that the workload requires from each of the devices in the processing facility
- "Job flow balance" assumption
  - The system is fast enough to handle the arrivals, and thus the completion rate or throughput equals to the arrival rate.

Example 6-2: Using Predicted Metric

Arrival rate, λ	0.4 jobs per sec
Mean service time, S	2 sec

By job flow balance assumption

We then calculate the following average values:

Throughput:	$X = \lambda$	= 0.4 jobs per sec
Utilization:	$U = XS$	= 0.4*2 = 0.8
Residence time:	$RT = S/(1-U)$	= 2/(1-0.8) = 10 sec
Queue length:	$N = X * RT$	=0.4*10 = 4 jobs

“Queue length” formula is fundamental to SPE. Next Slide!

Little’s Law

Queue length = Throughput * Residence time

(N = X * RT)

- The specifications and results from the simple model are average values
- However, in a mixed queue, individual results may differ
  - The metrics for a specific job may differ from the average
  - The difference depends on the distribution of arrivals and service requirements

Exercises: Use of Utilization Law

1) A network segment transmits 1,000 packets/sec. Each packet has an average transmission time equal to 0.15 msec. What is the utilization of LAN segment?

2) An NFS server was monitored during 30 minutes and the number of I/O operations performed during the period was found to be 10,800. The average number of active NFS requests was found to be three. What was the average response time per NFS request at the server?

Queueing Network Model (QNM)

- A model that encompasses a network of queues and servers is known as a ***queueing network model*** (QNM)
- Two types of QNM: ***closed models*** and ***open models***

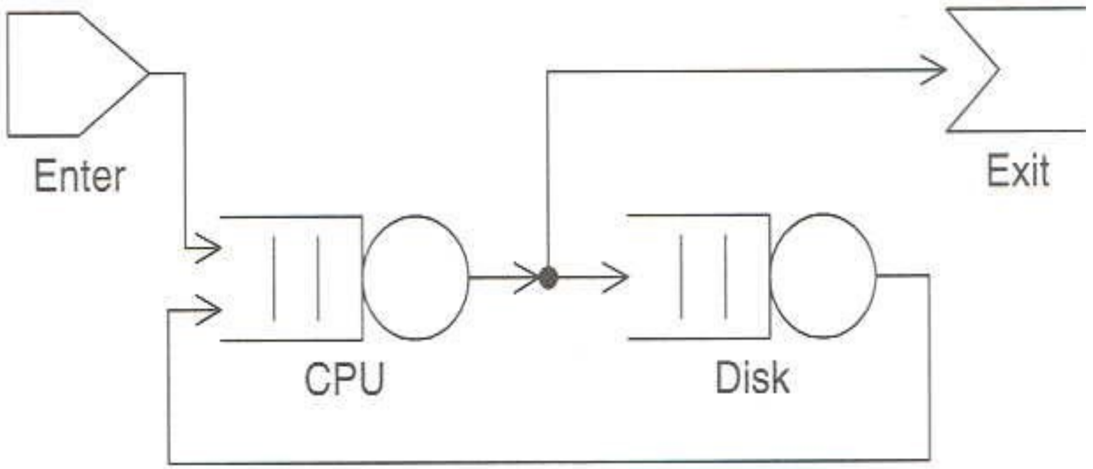


Figure 6-3: A Simple QNM

Open QNM

- Open QNM is appropriate for systems with external arrivals and departures, such as ATM
- For an open QNM, specify the ***workload intensity*** and ***service requirements***
- The workload is the ***arrival rate*** that rate at which jobs arrive for service
- The service requirements are ***the number of visits*** for each device, and ***the average service time per visit***, or ***the total demand*** for that device

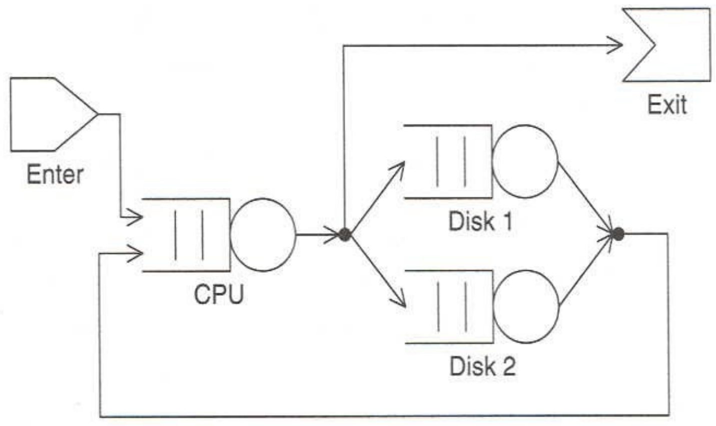


Figure 6-4: Open QNM

Example 6-3: Open QNM Computation

- First, we specify the following parameters:

λ	System arrival rate
V_i	Number of visits to device
S_i	Mean service time at device

- Next, we calculate the performance metrics using the following formulas:

1) System throughput (X_0): $X_0 = \lambda$

2) Throughput of device i (X_i): $X_i = X_0 \times V_i$

3) Utilization of device i (U_i): $U_i = X_i \times S_i$

4) Residence time per visit at device i (RT_i): $RT_i = \dfrac{S_i}{1 - U_i}$

5) Queue length for device i (N_i): $N_i = X_i \times RT_i$

6) System queue length (N): $N = \Sigma N_i$

7) System response time (RT): $RT = N / X_0$

Example 6-4: Open QNM Solution

- Sample parameters:

System arrival rate, $\lambda = 5$ jobs per second

Number of visits, V		**Mean service time, S**	
CPU	5	CPU	0.01
Disk1	3	Disk1	0.03
Disk2	1	Disk2	0.02

Metrics	**CPU**	**Disk1**	**Disk2**
1. **X**, throughput			
2. **S**, mean service time			
3. **U**, utilization			
4. **RT**, residence time			
5. **N**, queue length			
Total jobs in system =			
System response time =			

Example 6-4 (con’t)

Metrics	CPU	Disk1	Disk2
1. **X**, throughput	25	15	5
2. **S**, mean service time	0.01	0.03	0.02
3. **U**, utilization	0.25	0.45	0.10
4. **RT**, residence time	0.013	0.055	0.022
5. **N**, queue length	0.325	0.825	0.111
Total jobs in system = 0.325 + 0.825 + 0.111 = 1.26			
System response time = 1.26/5 = 0.252 sec			

Exercise

- Database transactions perform an average of 4.5 I/O operations on the database server. The database server was monitored during one hour and during this period, 7,200 transactions were executed during this period. What is the average throughput of the disk? If each disk I/O takes 20 msec on average, what was the disk utilization?

Closed QNM

- Closed QNM has no external arrivals or departures
- A fixed number of jobs keep circulating among queues

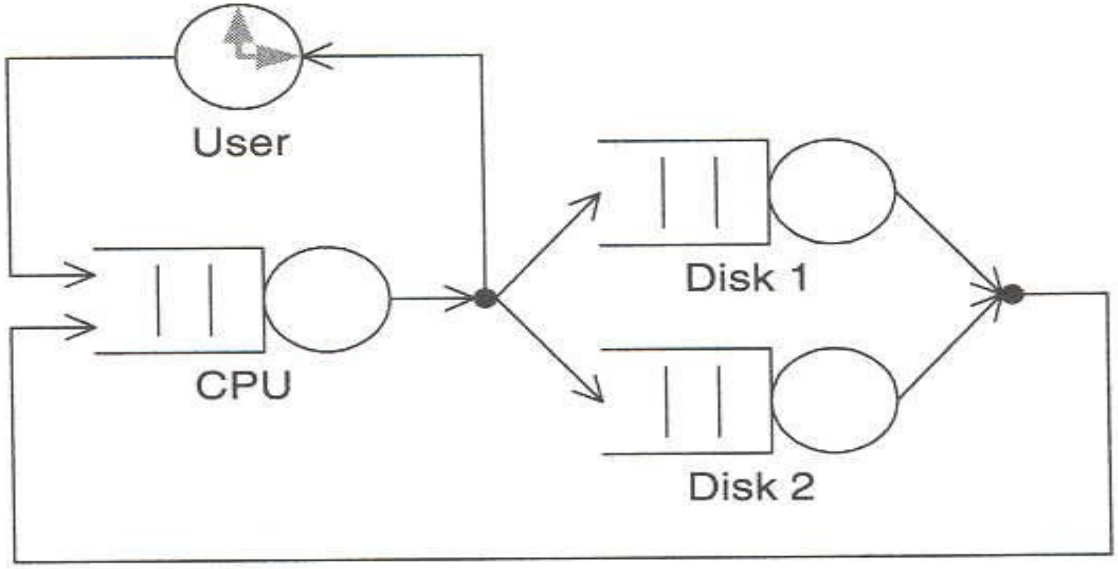


Figure 6-5: Closed QNM

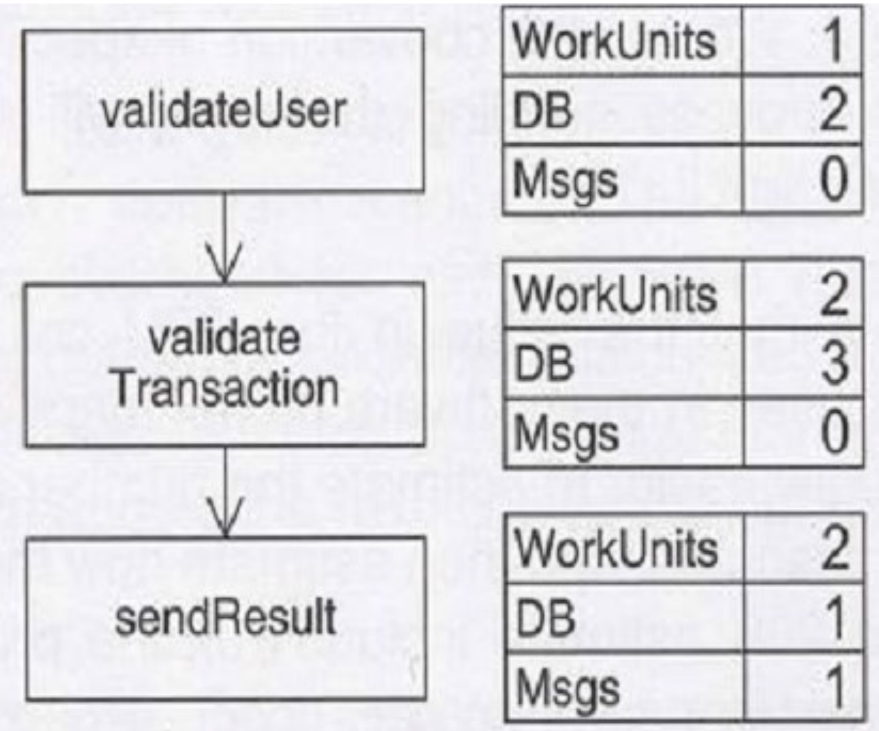
Solving Closed QNM (con't)

- This model needs
  - ***The number of users*** (or the number of simultaneous jobs)
  - ***The think time***, i.e., the average delay between the receipt of a response and the submission of the next
  - Number of visits
  - Service time
  - Total demand for each device

Deriving System Model Parameters from Software Model Results

- *Step 1:* use queue-servers to represent the key computer resources or devices that you specified in the software execution model and add the connections between queues to complete a model topology
- *Step 2:* decide whether the system is best modeled as an open or closed QNM
- *Step 3*: determine the workload intensities for each scenario
- *Step 4:* specify the service requirements

Example 6-5



WorkUnits	1
DB	2
Msgs	0

WorkUnits	2
DB	3
Msgs	0

WorkUnits	2
DB	1
Msgs	1

Software Model for authorizeTransaction

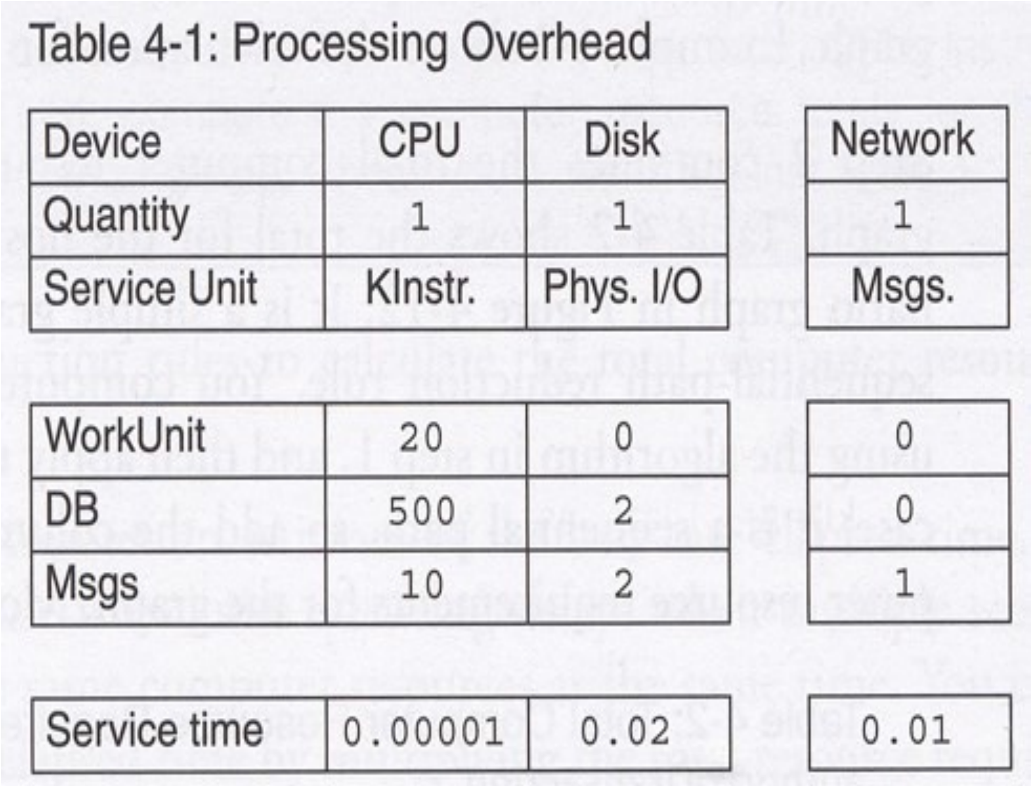
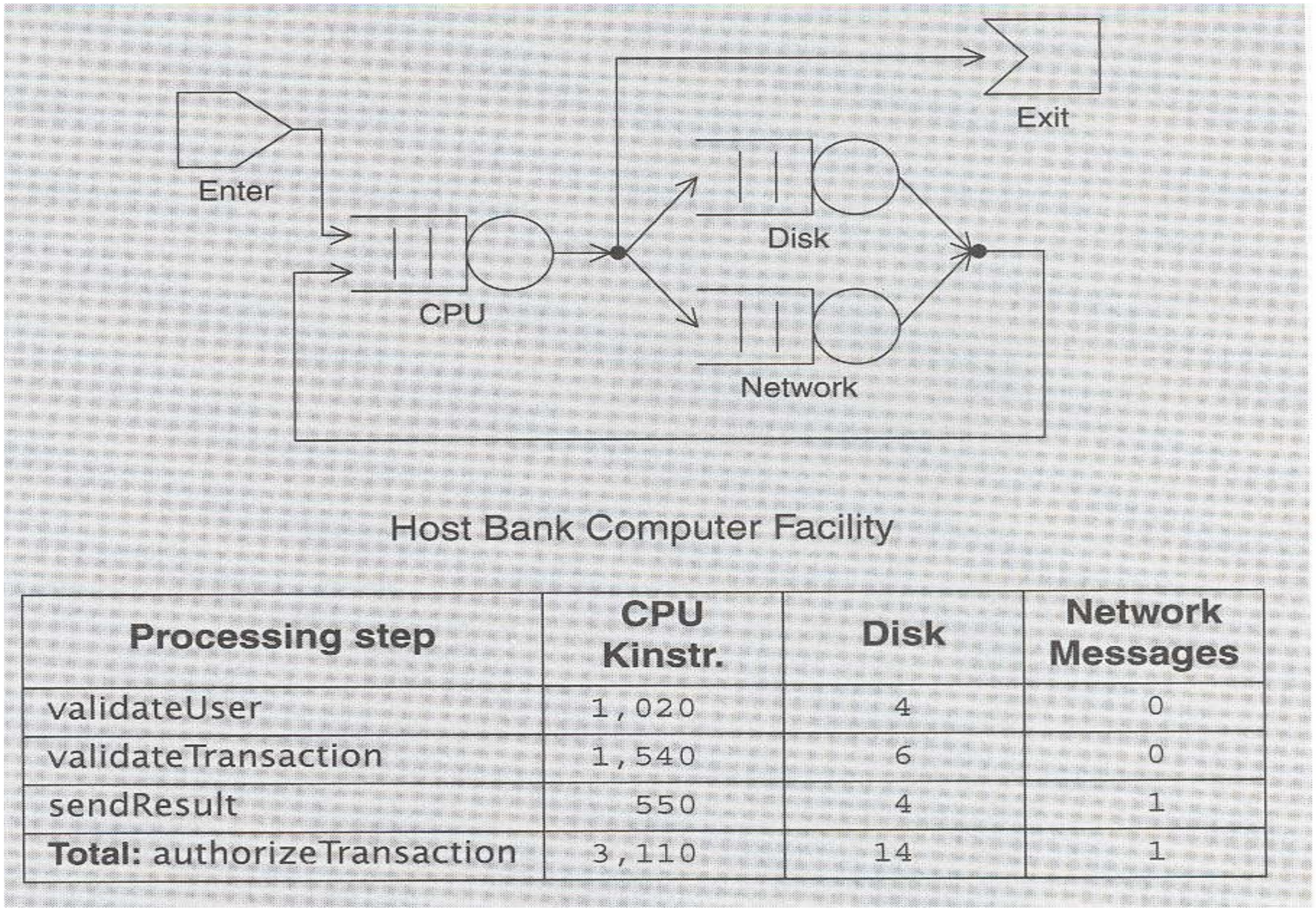


Table 4-1: Processing Overhead

Device	CPU	Disk	Network
Quantity	1	1	1
Service Unit	KInstr.	Phys. I/O	Msgs.
WorkUnit	20	0	0
DB	500	2	0
Msgs	10	2	1
Service time	0.00001	0.02	0.01

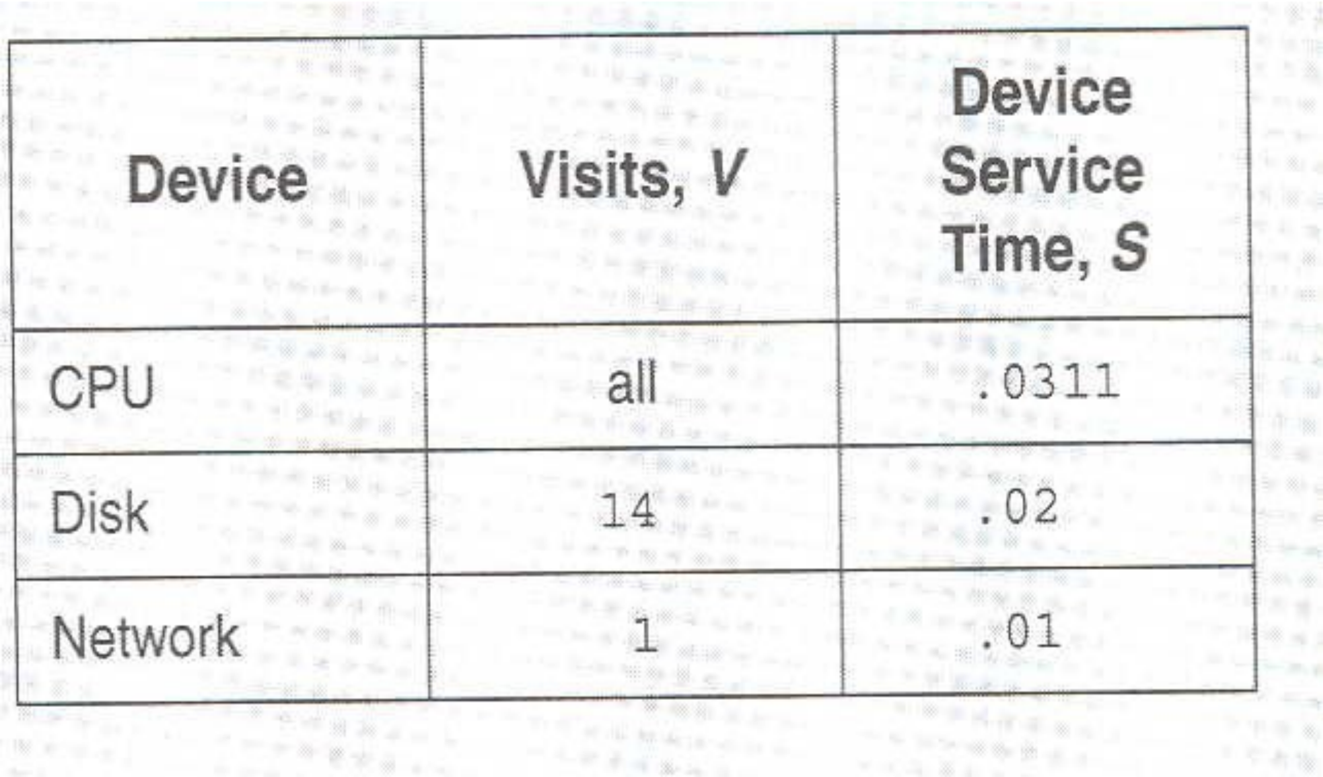
Example 6-5 (con't)



Host Bank Computer Facility

Processing step	CPU Kinstr.	Disk	Network Messages
validateUser	1,020	4	0
validateTransaction	1,540	6	0
sendResult	550	4	1
Total: authorizeTransaction	3,110	14	1

Example 6-5 (con’t)



Device	Visits, *V*	Device Service Time, *S*
CPU	all	.0311
Disk	14	.02
Network	1	.01

What System Execution Model Can’t do?

- Intricate details of computer system devices
- Passive resources: resource that are required for processing but do no work themselves, e.g., memory
- Additional metrics, such as minimum, maximum, and variance; arrival distributions
- It is possible to model the additional aspects of execution behavior using more advanced or alternative techniques if it is important

Distributed System Case Study

- The software supports an electronic virtual storefront
- This case study focuses the Customer Service component that
  - Collects completed order
  - Initiates tasks in the other components
  - Tracks the status of orders in progress

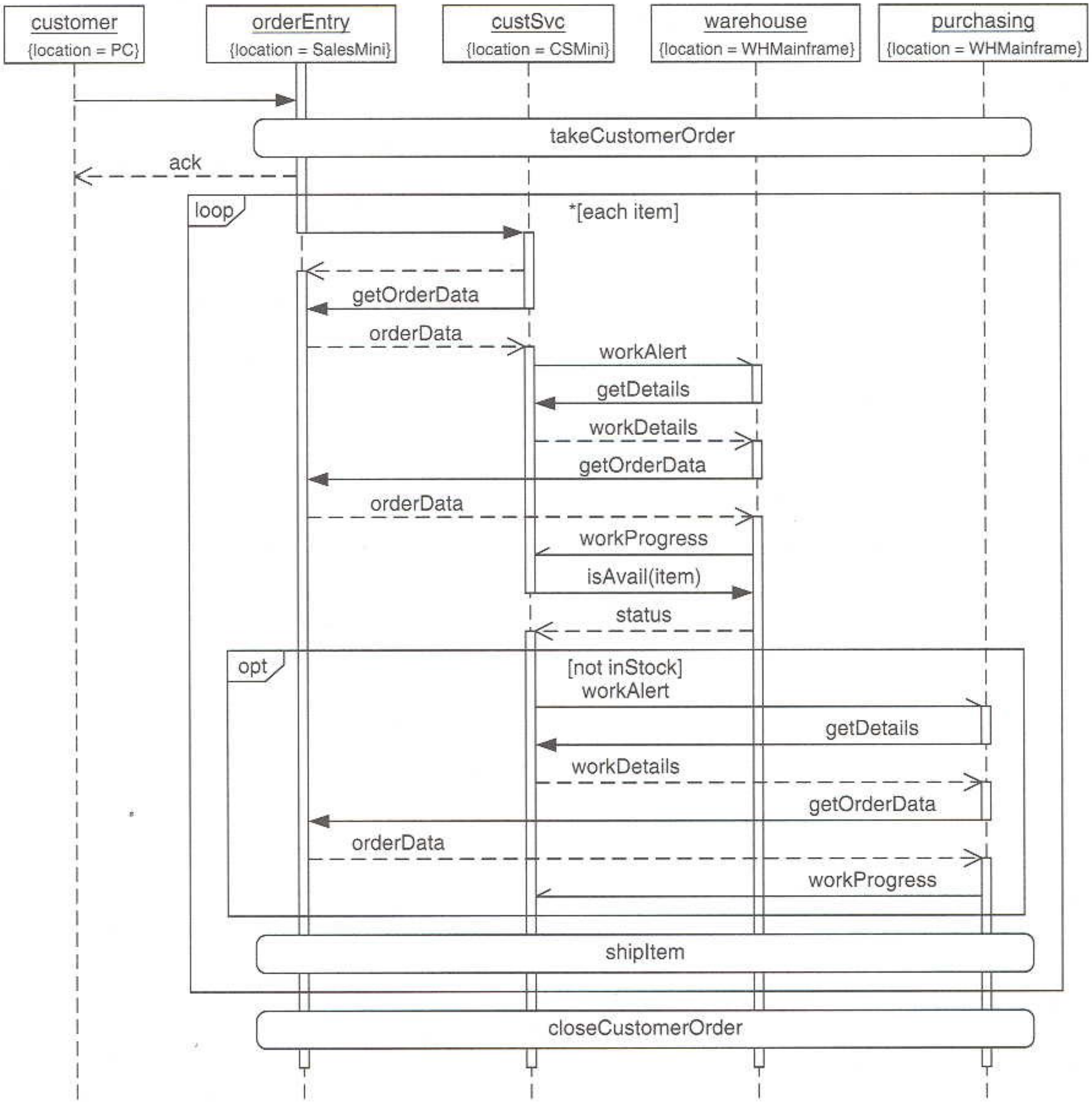
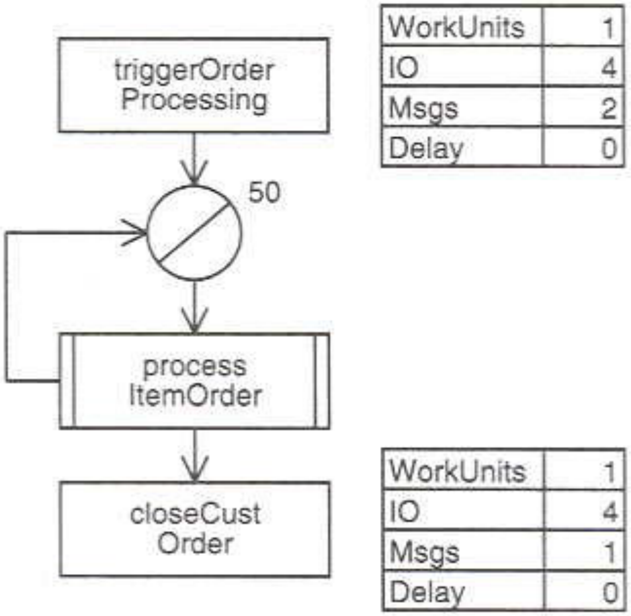


Figure 6-6: New Order Scenario



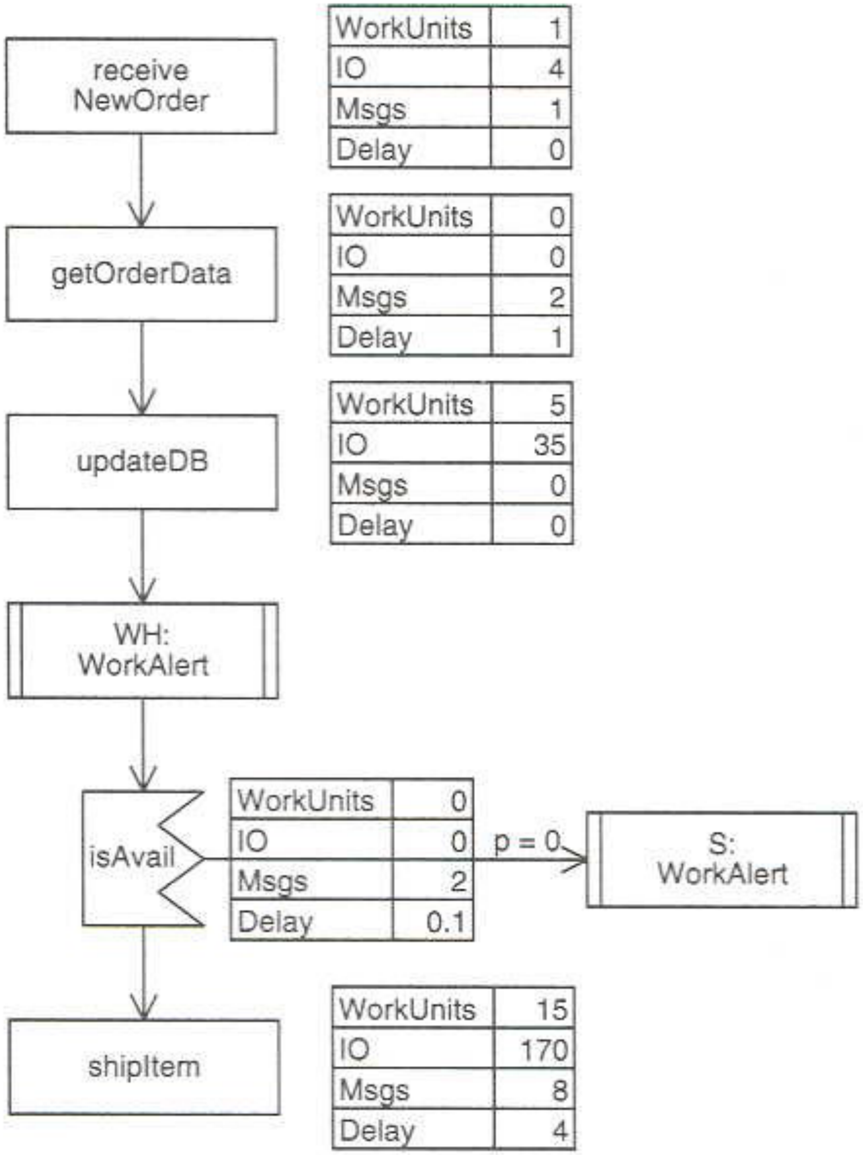
triggerOrder Processing

WorkUnits	1
IO	4
Msgs	2
Delay	0

closeCust Order

WorkUnits	1
IO	4
Msgs	1
Delay	0

Figure 6-7: Execution Graph custServ:NewOrder



receive NewOrder

WorkUnits	1
IO	4
Msgs	1
Delay	0

getOrderData

WorkUnits	0
IO	0
Msgs	2
Delay	1

updateDB

WorkUnits	5
IO	35
Msgs	0
Delay	0

isAvail

WorkUnits	0
IO	0
Msgs	2
Delay	0.1

shipItem

WorkUnits	15
IO	170
Msgs	8
Delay	4

Figure 6-8: Expansion of processItemOrder

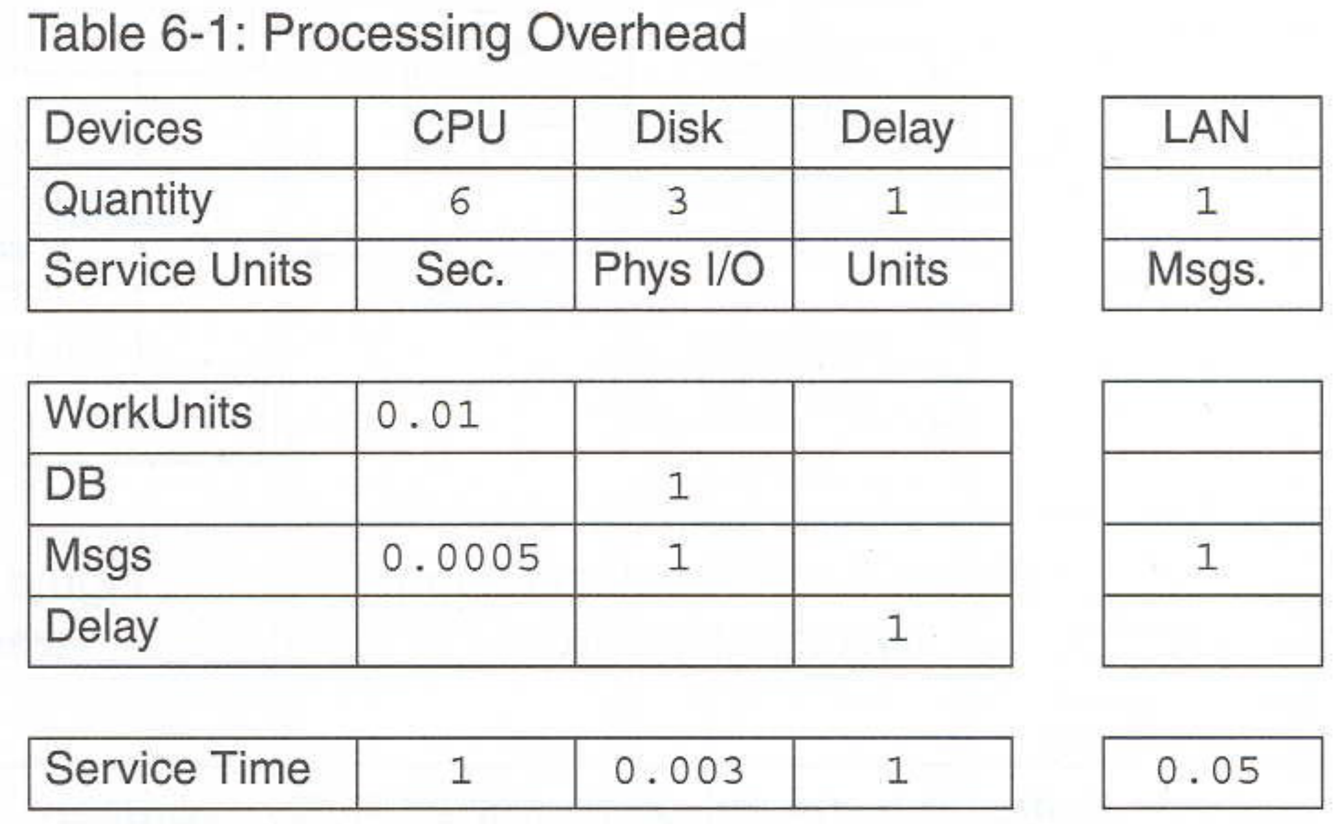


Table 6-1: Processing Overhead

Devices	CPU	Disk	Delay	LAN
Quantity	6	3	1	1
Service Units	Sec.	Phys I/O	Units	Msgs.

WorkUnits	0.01			
DB		1		
Msgs	0.0005	1		1
Delay			1	

Service Time	1	0.003	1	0.05

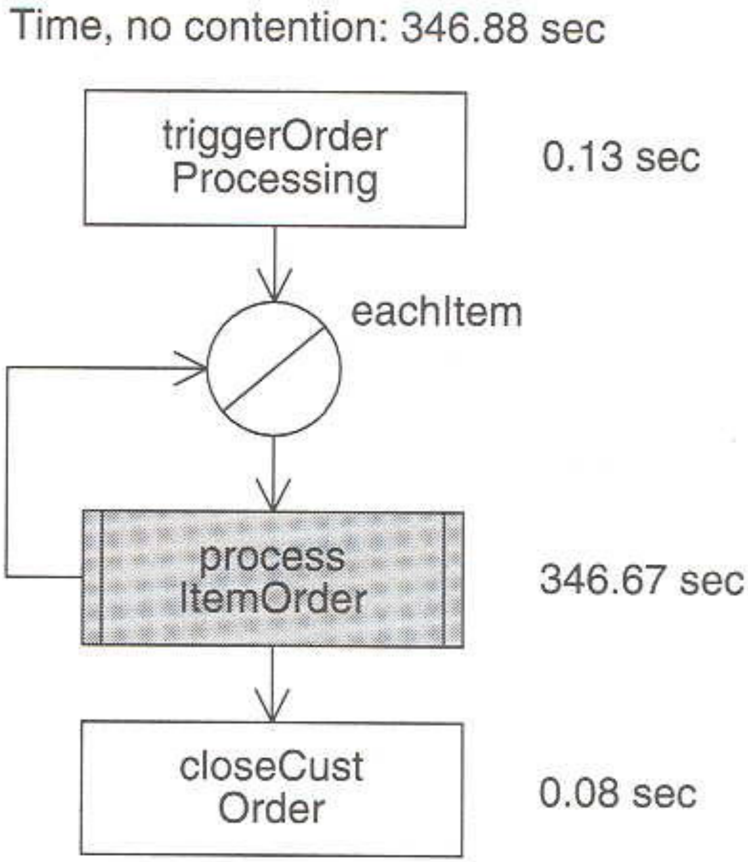


Figure 6-9: Best-Case Elapsed Time for newOrder Scenario

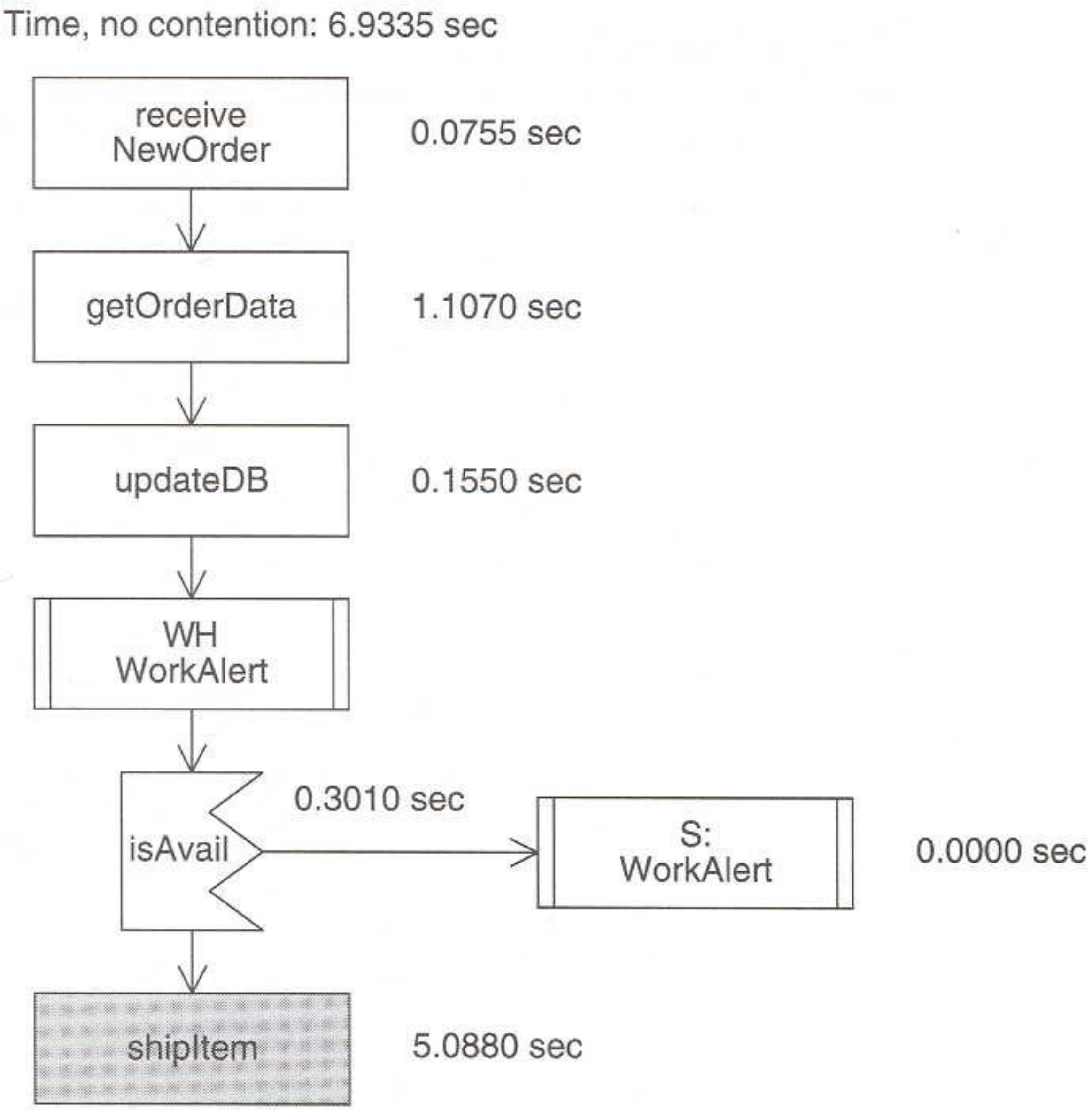


Figure 6-10: Best-case Elapsed Time for Each processItemOrder

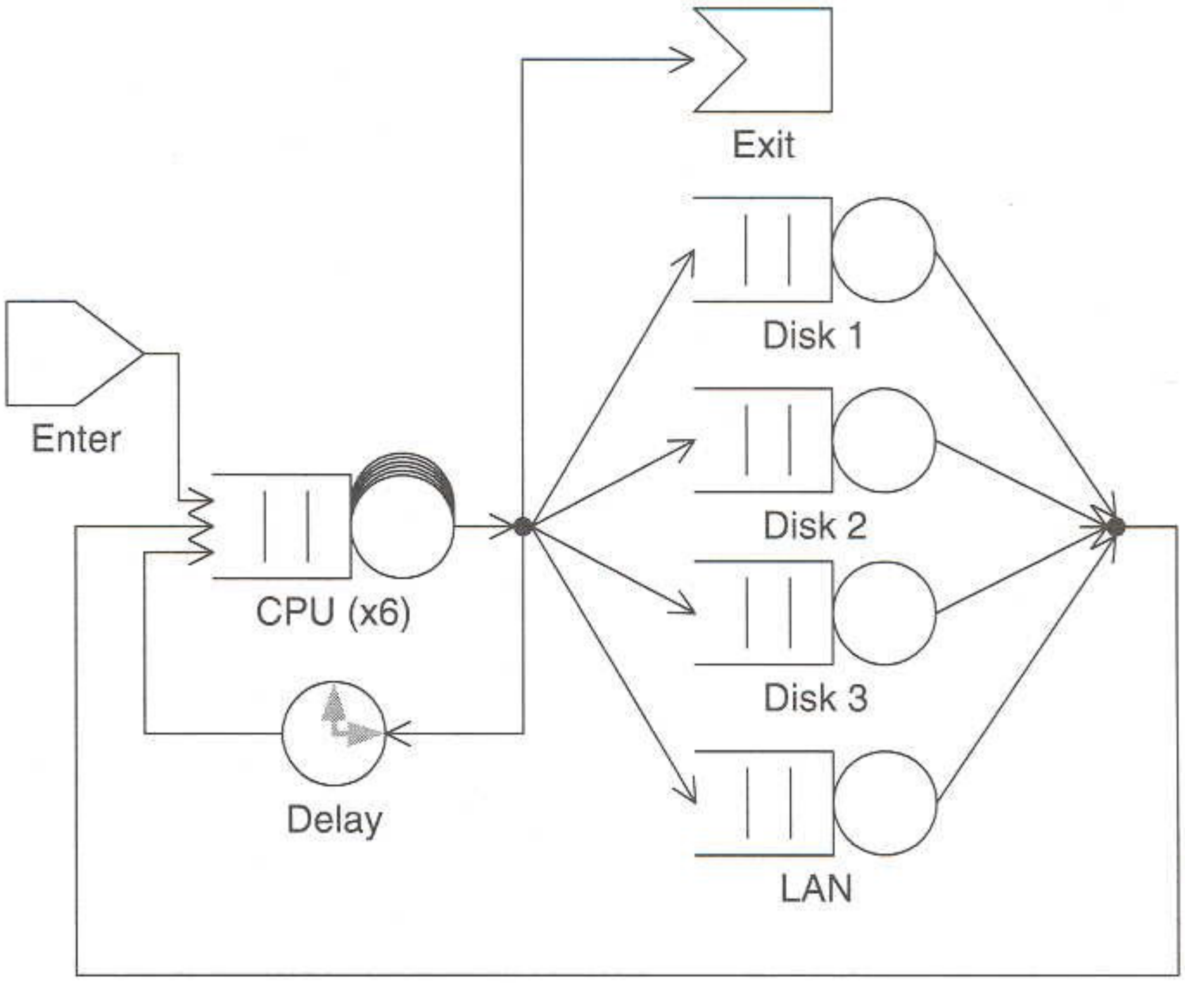
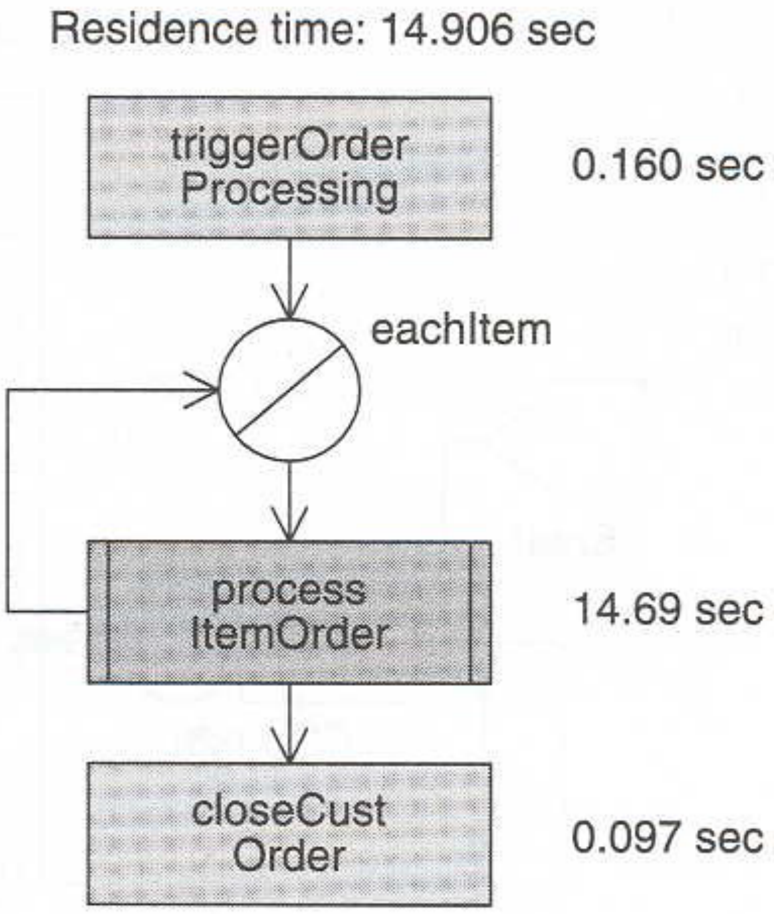
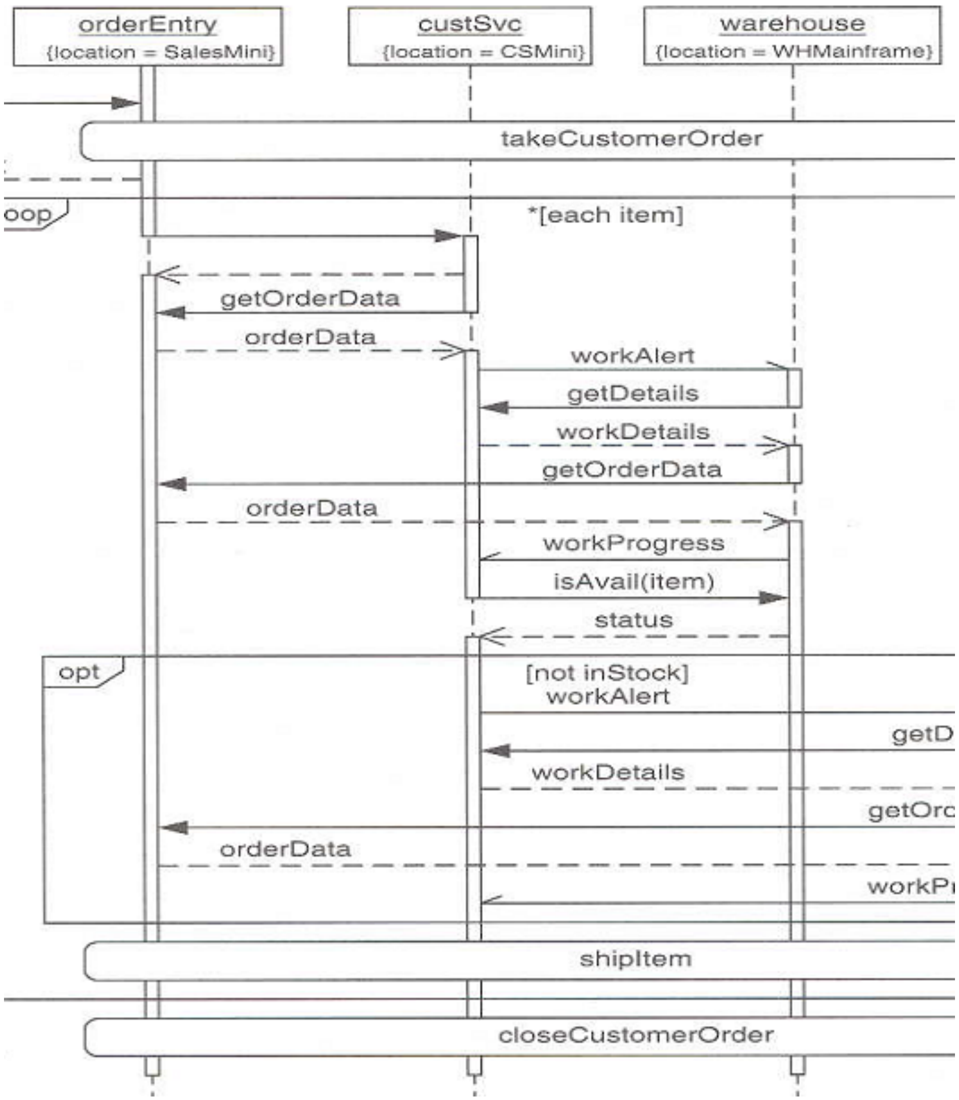


Figure 6-11: QNM for the Customer Service Facility



re 6-12: System Model Results for Grouped Items



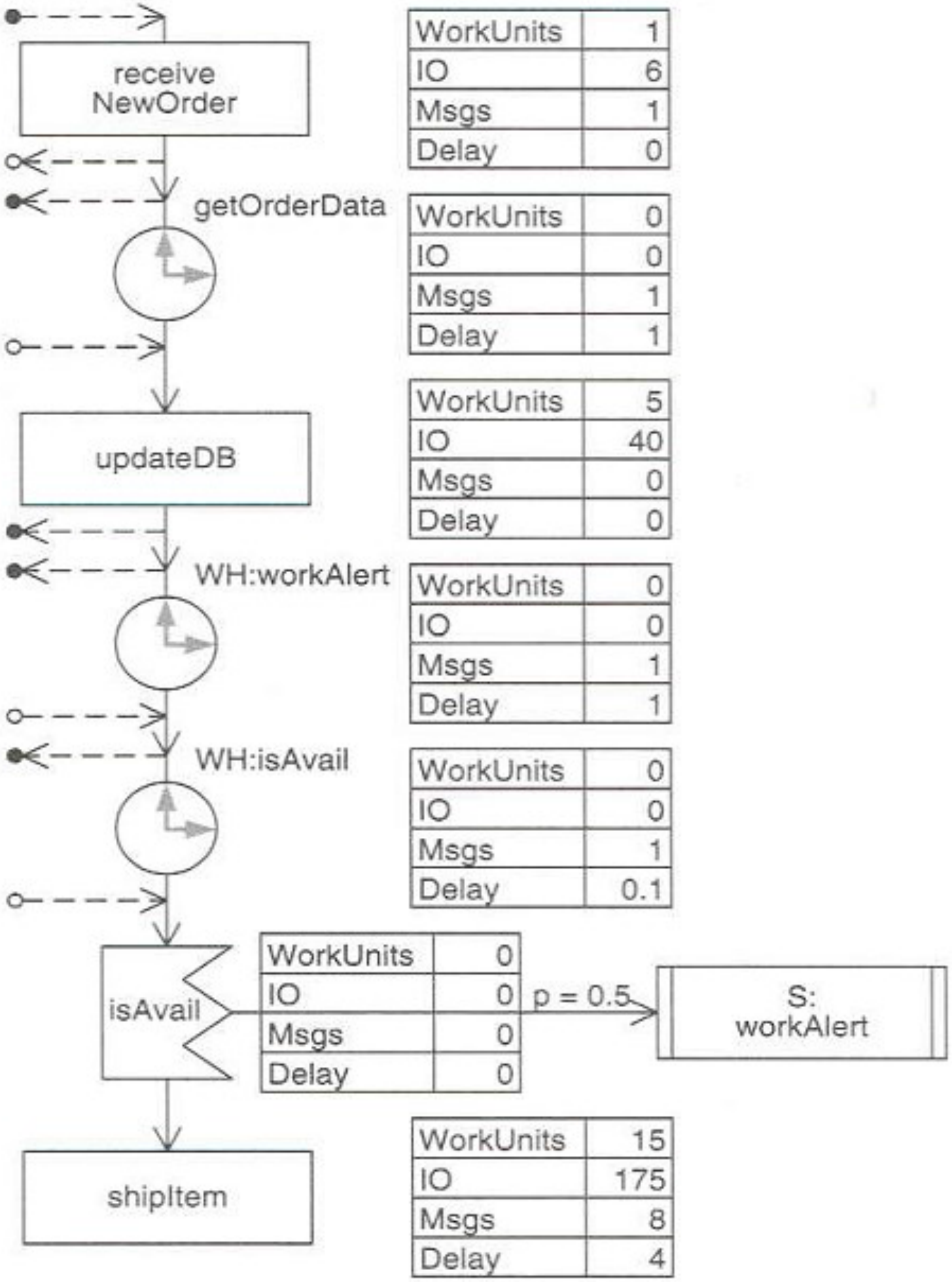
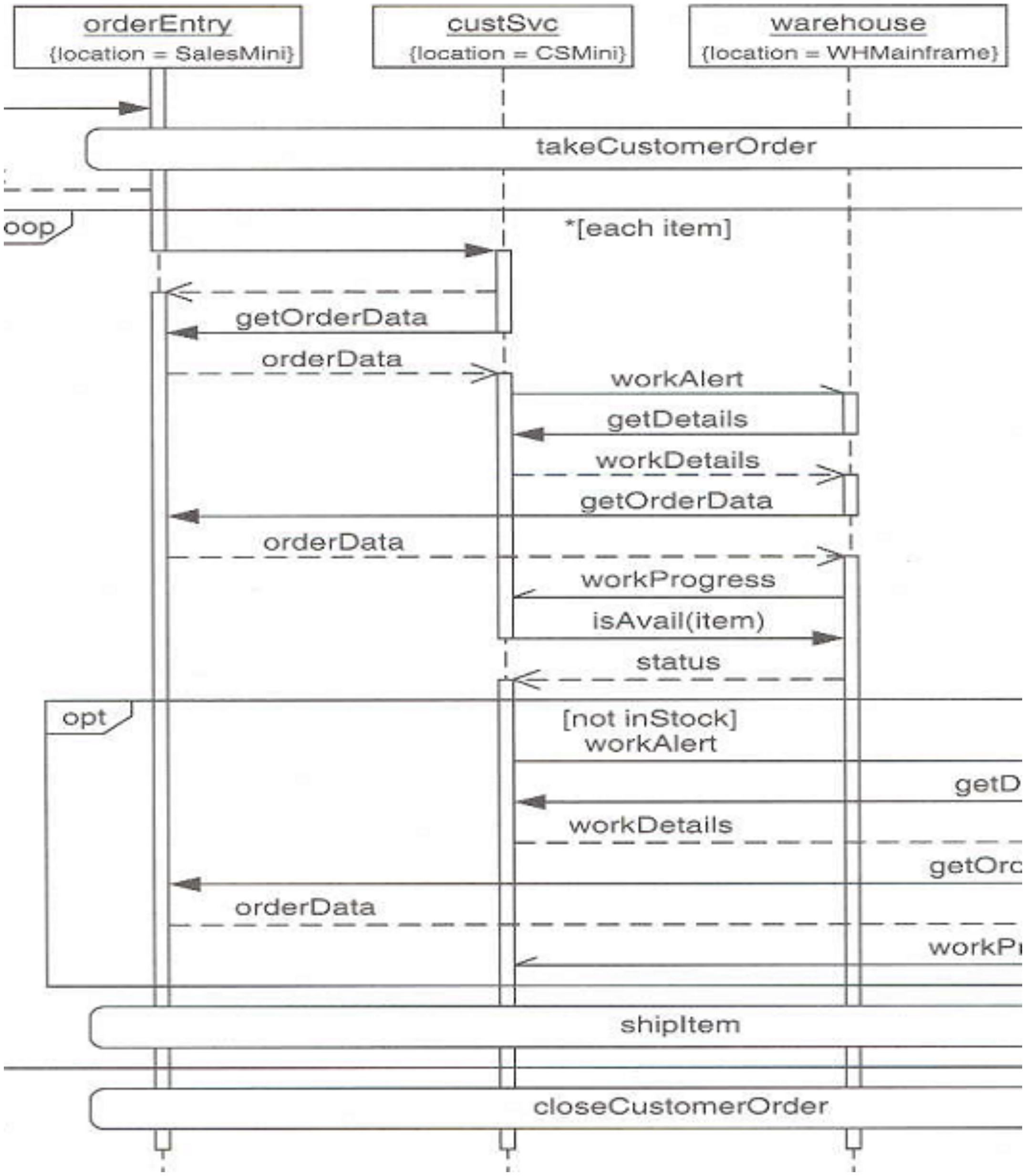


Figure 6-13: Synchronization in processItemOrder



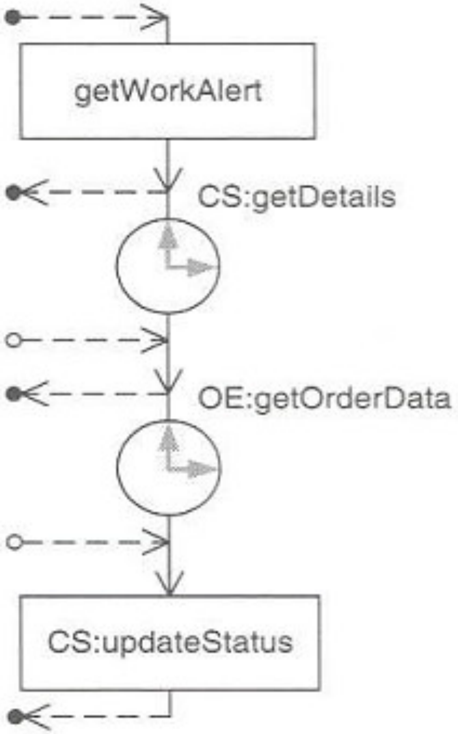


Figure 6-14: WH:work Alert Processing

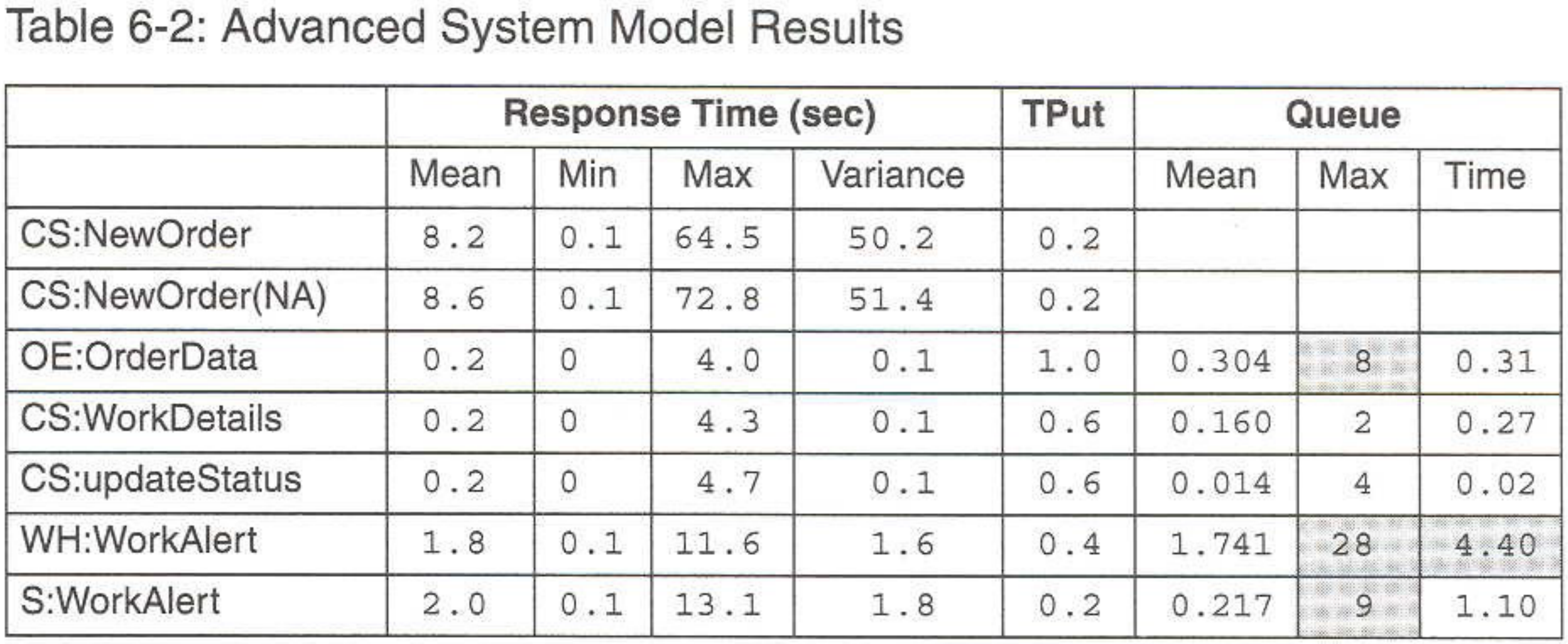


Table 6-2: Advanced System Model Results

	Response Time (sec)				TPut	Queue		
	Mean	Min	Max	Variance		Mean	Max	Time
CS:NewOrder	8.2	0.1	64.5	50.2	0.2			
CS:NewOrder(NA)	8.6	0.1	72.8	51.4	0.2			
OE:OrderData	0.2	0	4.0	0.1	1.0	0.304	8	0.31
CS:WorkDetails	0.2	0	4.3	0.1	0.6	0.160	2	0.27
CS:updateStatus	0.2	0	4.7	0.1	0.6	0.014	4	0.02
WH:WorkAlert	1.8	0.1	11.6	1.6	0.4	1.741	28	4.40
S:WorkAlert	2.0	0.1	13.1	1.8	0.2	0.217	9	1.10

Modeling Hints

- Multiple users and workload (e.g., arrival rate, the number of users, and think time)
- Average vs. Peak Performance
  - Basis QNMs calculates average values
- Sensitivity: if a small change in one parameter causes a large change in the computed metrics, the model is sensitive to that quantity
- Scalability: improves response times for your anticipated future loads
- Bottlenecks: the bottleneck device is the one with the highest utilization