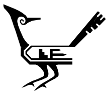
Software Performance Models for Distributed Systems

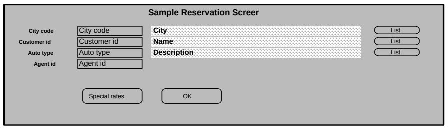
Overview

- ❖ SPE approach for early assessments of distributed systems
- ❖ Approximation approach
- ❖ Modeling synchronization and coordination
  - ✧ Approximate analytic solution
  - ✧ Simulation solution
- ❖ Case study

Part 1: Distributed System Performance Approximation



Sample Screen



Sample Reservation Screen

City code: City code

Customer id: Customer id

Auto type: Auto type

Agent id: Agent id

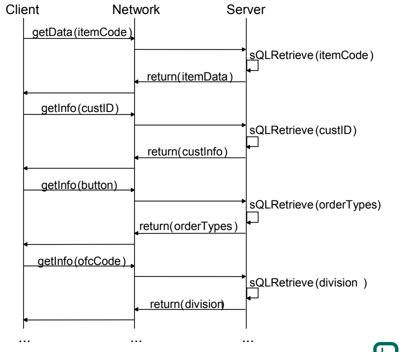
City

Name

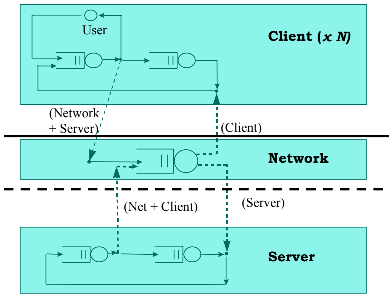
Description

Processing steps?

Sequence Diagram



Partition the Distributed System Model

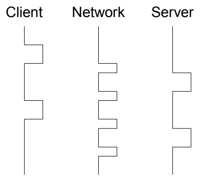


- •Identify primary facility
- •Construct software/system model
- •Estimate delays for other system interactions
- •Iterative solutions refine estimates

+ Firewalls + Web/Application/Database Servers + Internal Network

Early Life Cycle Models

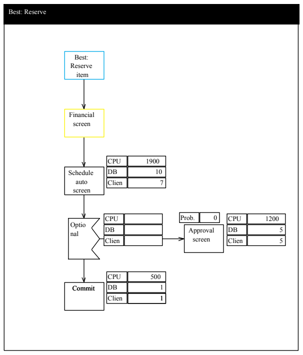
- Focus on one scenario/processor at a time
- Approximate delay for "blips" in other scenarios



Distributed System Model Solutions

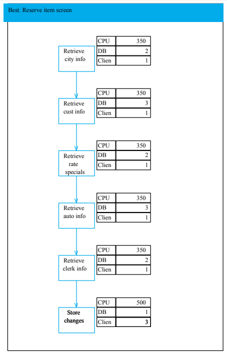
- 1. Use software execution model approximation techniques for estimating the response time of remote calls
- 2. Use iterative solutions – solve for response time of remote calls and substitute for delay estimates
- 3. Use "advanced system execution model" simulation to study the effect of queueing, contention for shared processes, and other delays for inter-process coordination.

1. Server Software Model

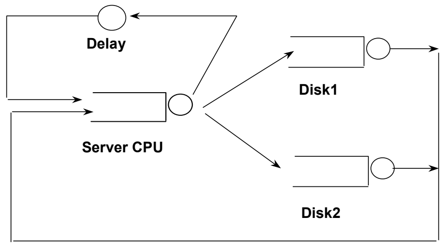


CPU?
I / Os?
Delay?

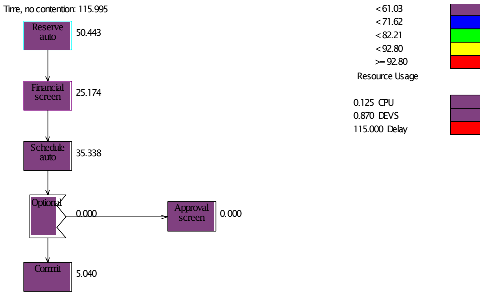
Reserve Item Screen



Server QNM

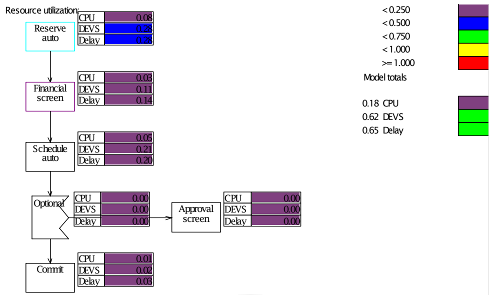


Results



Best case end-to-end response time, 116 sec., is too high.

More Results



Disk utilization 62% is too high for best case.

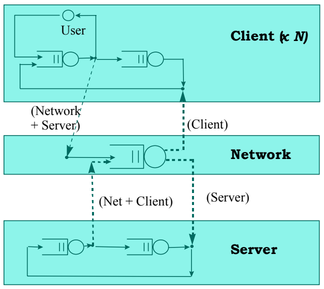
Case Study Results

- Detected end-to-end response time problem
- Initial configuration sizing problems
- Performance benchmarks provided initial model data

Client Model

- Scenario (s)?
  - In this example, the processing steps are the same
- Software resource requirements?
  - The resource requirements reflect client processing rather than server processing
- Delay time?
  - The server model solution provides an estimate for client delay
- Client QNM?
  - Represent client devices and a delay server for other processing(only)

"Simple" Combined Model



Multiple clients?

Multiple servers?

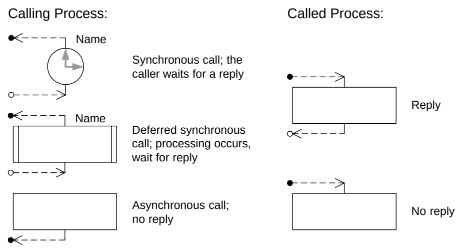
Multiple workloads?

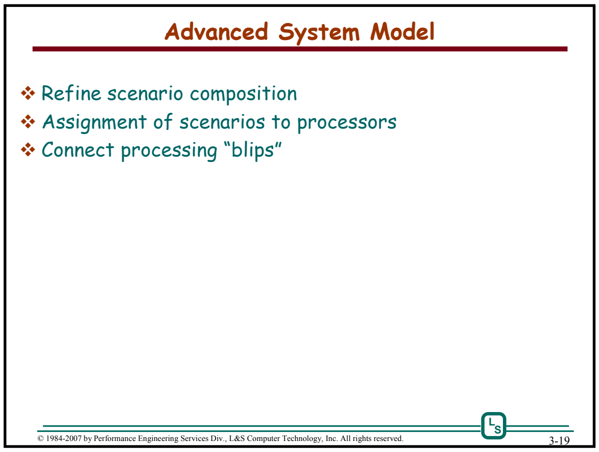
Servers connected with WANs?

Defer the combined model until later in development.

Part 2: Modeling Synchronization and Coordination

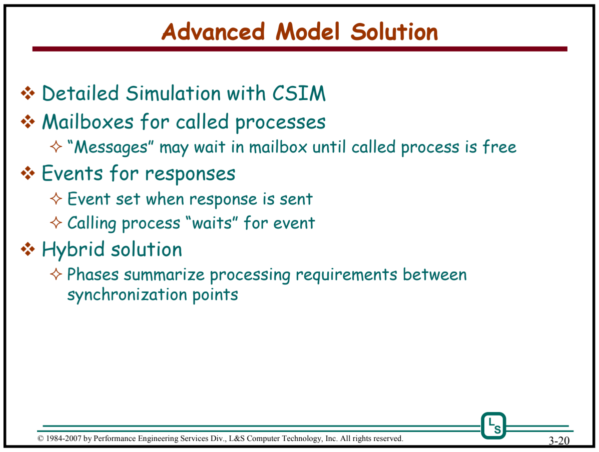
Execution Graph Synchronization Nodes


Calling Process:
Called Process:
Name
Synchronous call; the caller waits for a reply
Reply
Name
Deferred synchronous call; processing occurs, wait for reply
Asynchronous call; no reply
No reply



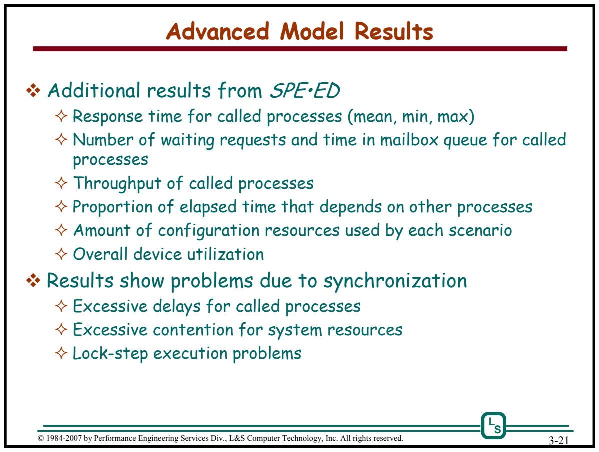
Advanced System Model

- ❖ Refine scenario composition
- ❖ Assignment of scenarios to processors
- ❖ Connect processing "blips"



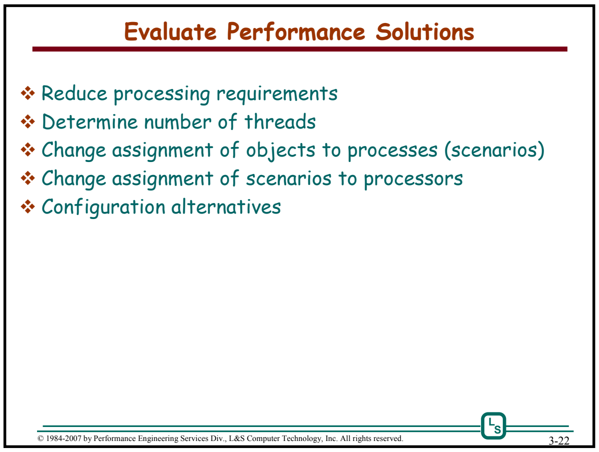
Advanced Model Solution

- ❖ Detailed Simulation with CSIM
- ❖ Mailboxes for called processes
  - ✧ "Messages" may wait in mailbox until called process is free
- ❖ Events for responses
  - ✧ Event set when response is sent
  - ✧ Calling process "waits" for event
- ❖ Hybrid solution
  - ✧ Phases summarize processing requirements between synchronization points



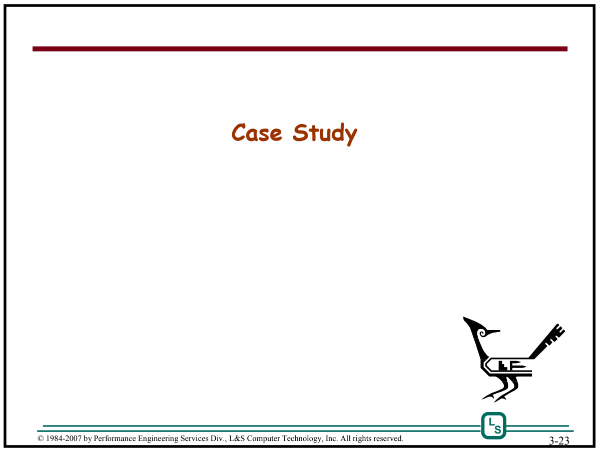
Advanced Model Results

- ❖ Additional results from *SPE•ED*
  - ✧ Response time for called processes (mean, min, max)
  - ✧ Number of waiting requests and time in mailbox queue for called processes
  - ✧ Throughput of called processes
  - ✧ Proportion of elapsed time that depends on other processes
  - ✧ Amount of configuration resources used by each scenario
  - ✧ Overall device utilization
- ❖ Results show problems due to synchronization
  - ✧ Excessive delays for called processes
  - ✧ Excessive contention for system resources
  - ✧ Lock-step execution problems

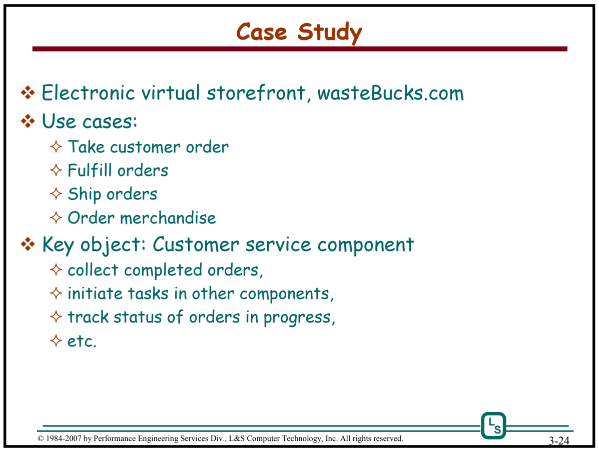


Evaluate Performance Solutions

- ❖ Reduce processing requirements
- ❖ Determine number of threads
- ❖ Change assignment of objects to processes (scenarios)
- ❖ Change assignment of scenarios to processors
- ❖ Configuration alternatives



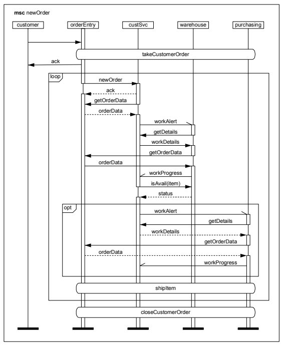
Case Study



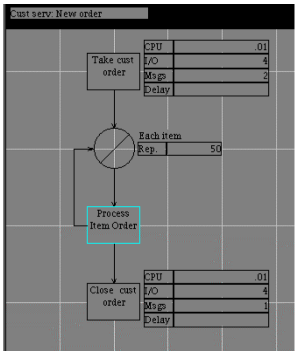
Case Study

- ❖ Electronic virtual storefront, wasteBucks.com
- ❖ Use cases:
  - ✧ Take customer order
  - ✧ Fulfill orders
  - ✧ Ship orders
  - ✧ Order merchandise
- ❖ Key object: Customer service component
  - ✧ collect completed orders,
  - ✧ initiate tasks in other components,
  - ✧ track status of orders in progress,
  - ✧ etc.

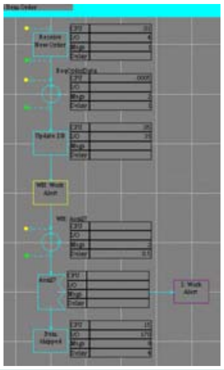
NewOrder Scenario



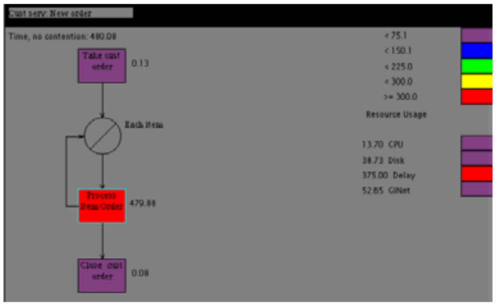
New Order Execution Graph



Process Item Order

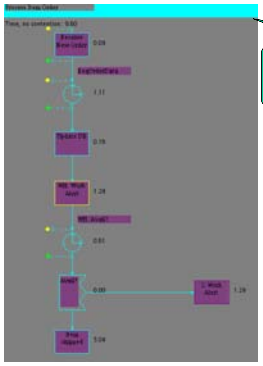


Best Case Results



Best case End-to-end response time is 480 sec.

Process Item Order Results (per item)

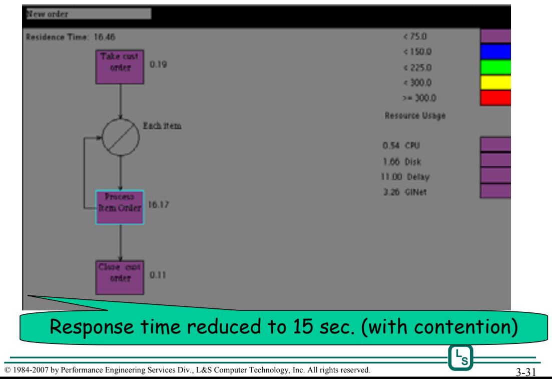


Best case elapsed time is 9.8 sec. per item

Performance Improvement

- Of the 9.8 seconds per item, 7.5 is estimated delay for remote processing
- Network congestion limits throughput
- Alternative selected that processes work orders as a group rather than individual items
- Model changes minor:
  - Number of repetitions -> 2 (ready order + back order)
  - Resource requirements for groups rather than items
  - Message time increases -> .06 (larger messages)

Results - 0.1 jobs/second



Response time reduced to 15 sec. (with contention)

Results

- ❖ Overhead and delays were significant portion of end-to-end time
- ❖ Architectural changes made significant improvement
- ❖ Simple models provide sufficient information for architectural evaluation
- ❖ Easy to formulate models and evaluate alternatives
- ❖ Resolve key performance problems before proceeding

Advanced System Model Solution

1. Simulation solution connects the processing across facilities:
   - ✧ If called process is busy, calling process queues
2. Results:
   - ✧ mean response time for called processes including time in queue (min & max)
     - ❍ Excessive dependent processing?
   - ✧ mean, variance, min, max number of requests in queue for called processes
     - ❍ Multi-threading?
   - ✧ throughput
     - ❍ Performance goal?

Advanced Model Results

	Response Time (secs.)				TPut	Queue		
	Mean	Min	Max	Variance		Mean	Max	Time
CS:NewOrder	14.4	0.8	72.7	79.51	.1			
OE:OrderData	0.16	0	2.6	0.05	.5	0.092	5	0.19
CS:WorkDetails	0.2	0	3.7	0.05	.3	0.057	2	0.19
CS:UpdStatus	0.1	0	4.4	0.04	.3	0.004	3	0.01
WH:WorkAlert	1.3	0	9.1	1.14	.2	0.122	9	0.62
P:WorkAlert	1.4	0	9.3	1.16	.1	0.019	3	0.193

- ❖ Max queue length & queue time suggest more threads for WH: WorkAlert for scalability
- ❖ Could show potential "lock-step" problems:
  - P: WorkAlert response time slightly higher than WH:WorkAlert even though throughput is lower
- ❖ The most important results came from early models!

Case Study Conclusions

- ❖ Important to assess distributed systems early
- ❖ Model progression important
  - ✧ Simple approximate models for early life cycle assessment
  - ✧ Advanced models for realistic projections and details of interconnection performance
- ❖ Techniques apply to other types of distributed systems such as Web Services and other middleware products

Summary

- ❖ SPE approach for early assessments of distributed systems
- ❖ Approximation approach
- ❖ Modeling synchronization and coordination
  - ✧ Approximate analytic solution
  - ✧ Simulation solution
- ❖ Case study