

# Planning the Capacity of a Web Server: An Experience Report

Daniel A. Menascé  
George Mason University  
menasce@cs.gmu.edu

Robert Peraino  
George Mason University  
peraino@gmu.edu

Nikki Dinh  
SRA International, Inc.  
nikki\_dinh@sra.com

Quan T. Dinh  
Lockheed Martin Federal Systems  
quan.t.dinh@lmco.com

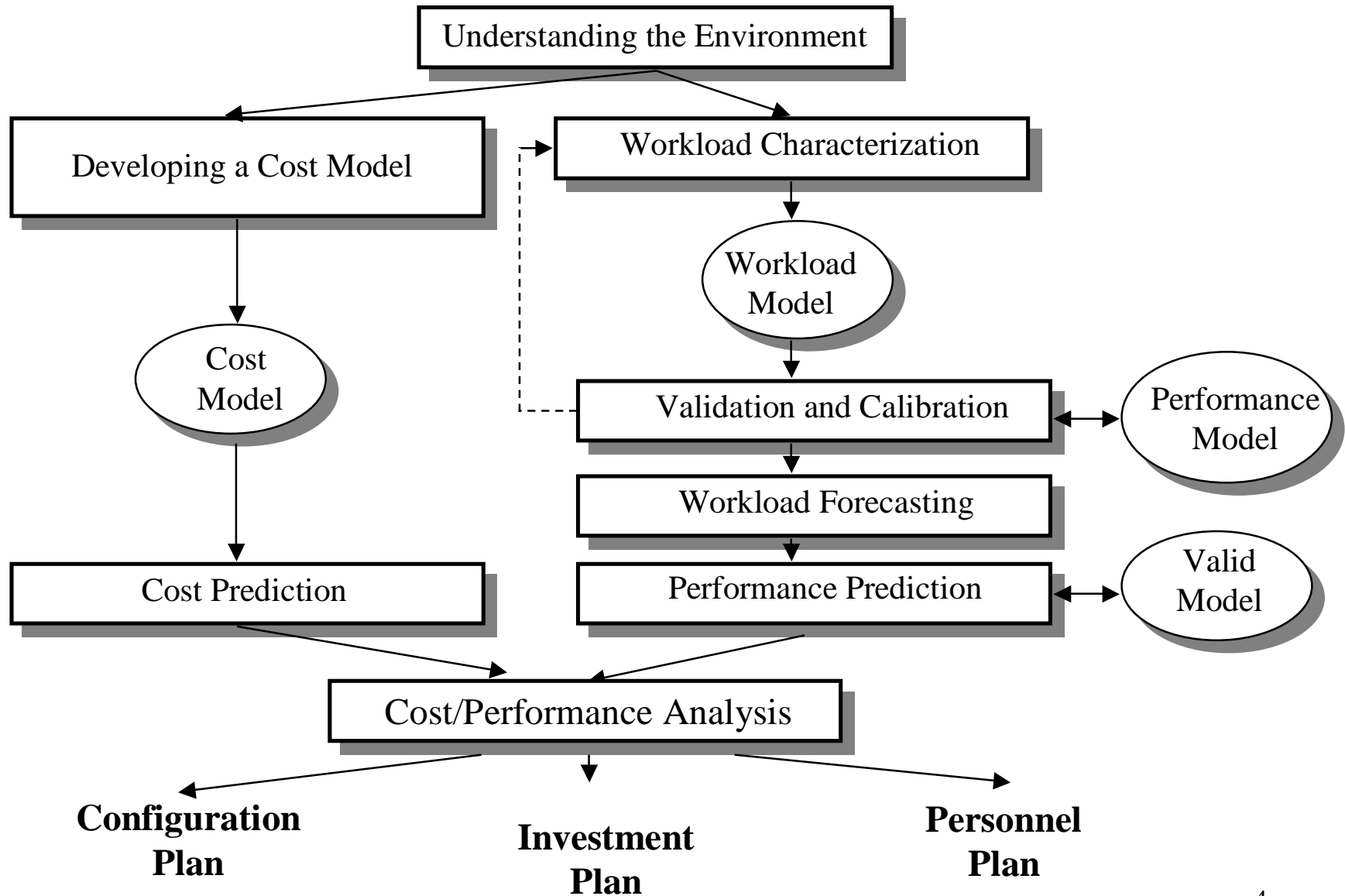
# Outline

- Motivation
- Capacity Planning Methodology
- Applying the Methodology
- Concluding Remarks

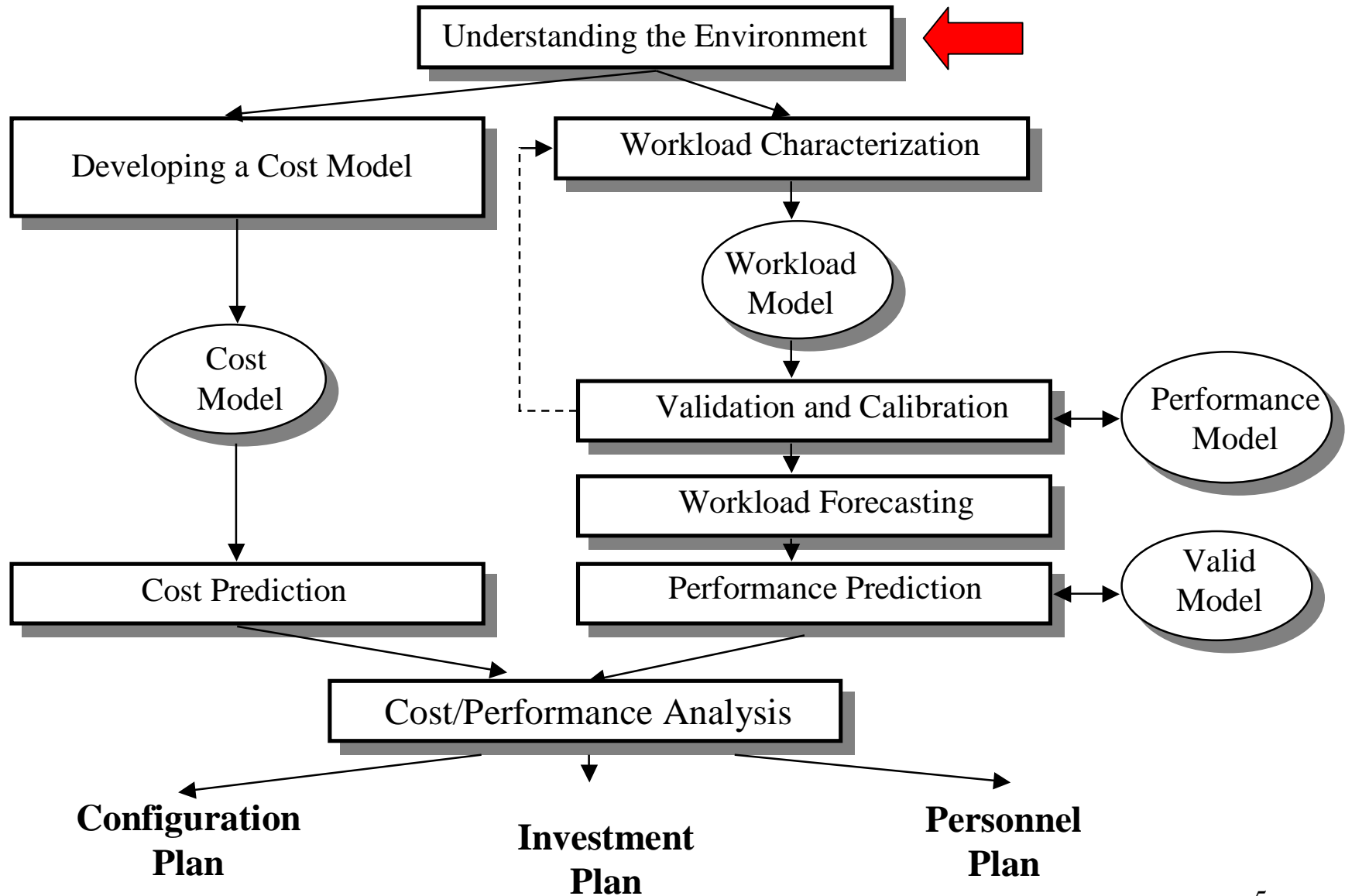
# Motivation

- University's WWW server is a highly visible production server.
- Server now supports a wide range of business processes.
- Lack of methodology to understand workload and size web server.

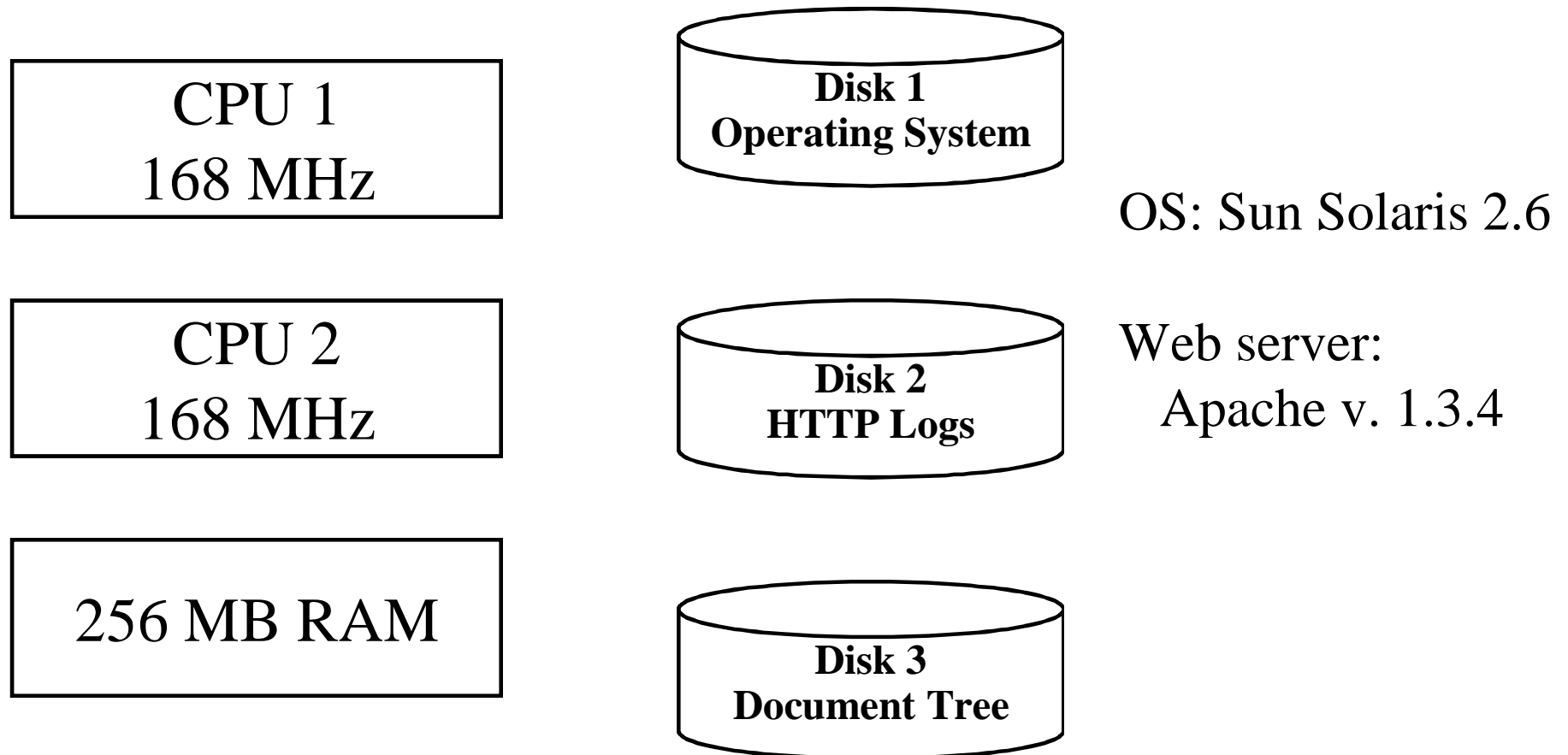
# Methodology



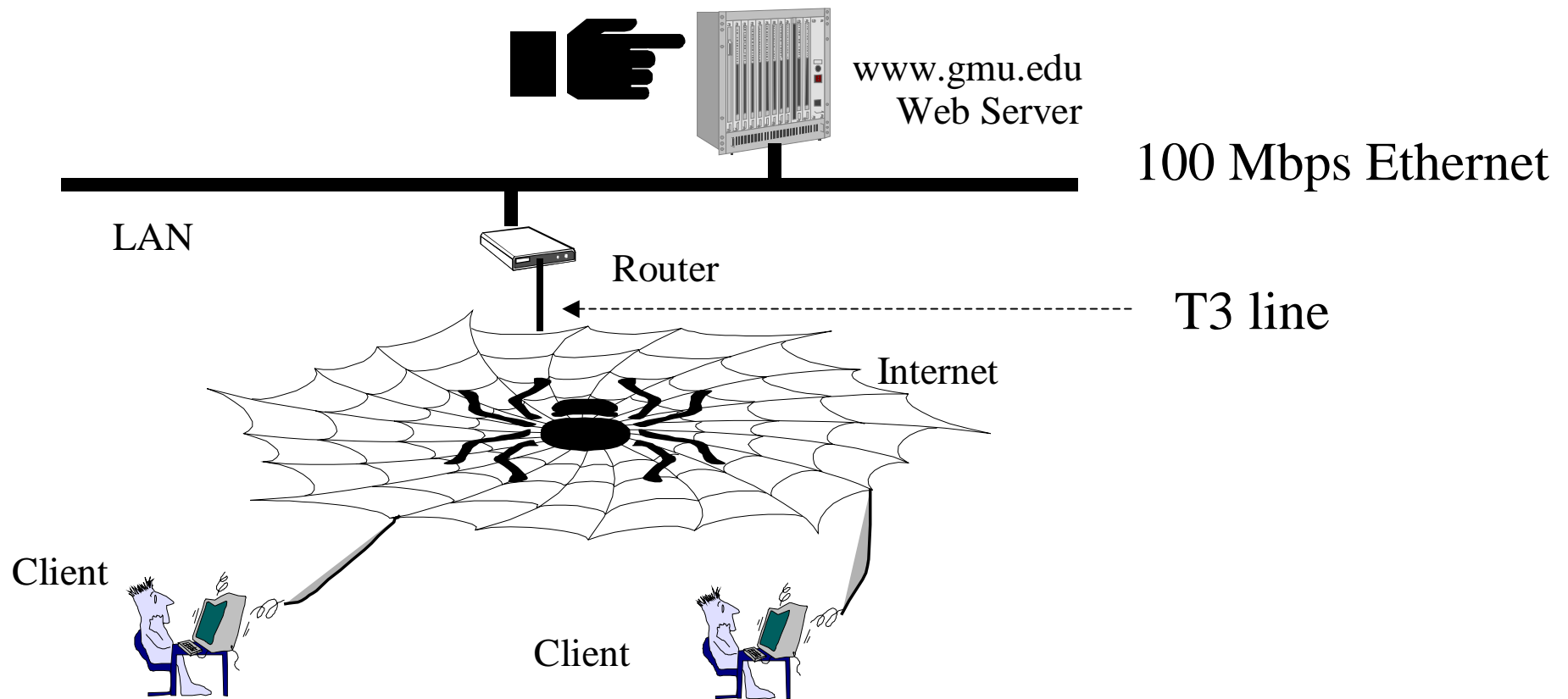
# Methodology



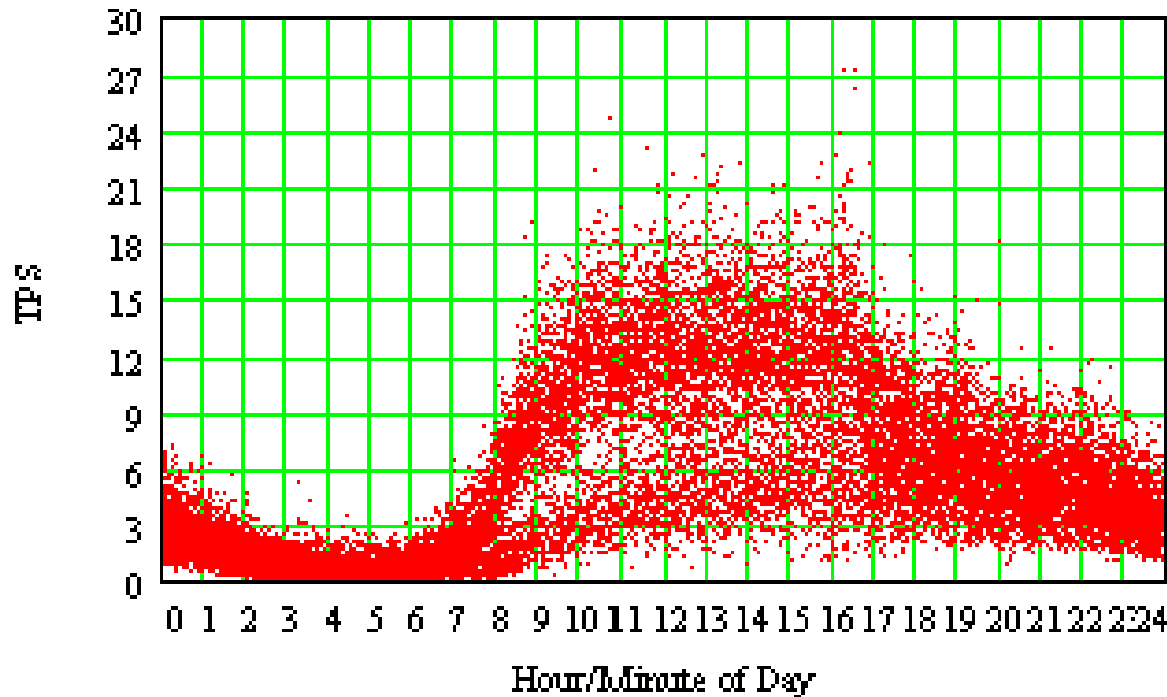
# Understanding the Environment: Server Configuration



# Understanding the Environment: Server Connectivity



# Understanding the Environment: Peak Period Analysis: Hour of Day



13.2 million request  
during February'99.  
Avg. computed for  
each minute.

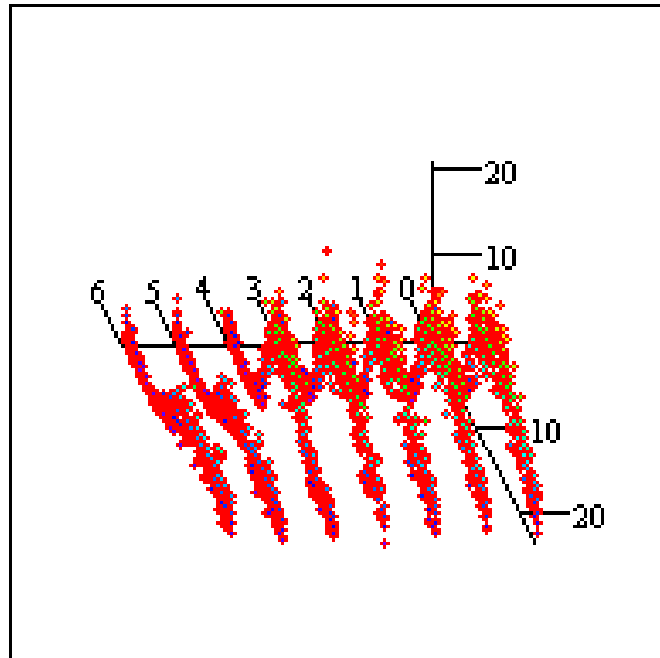
Peak period: 9 AM to 5 PM

Avg. arrival rate over entire month: 5.49 requests/sec

Maximum rate: 28.4 requests/sec

Minimum rate: 0.017 requests/sec

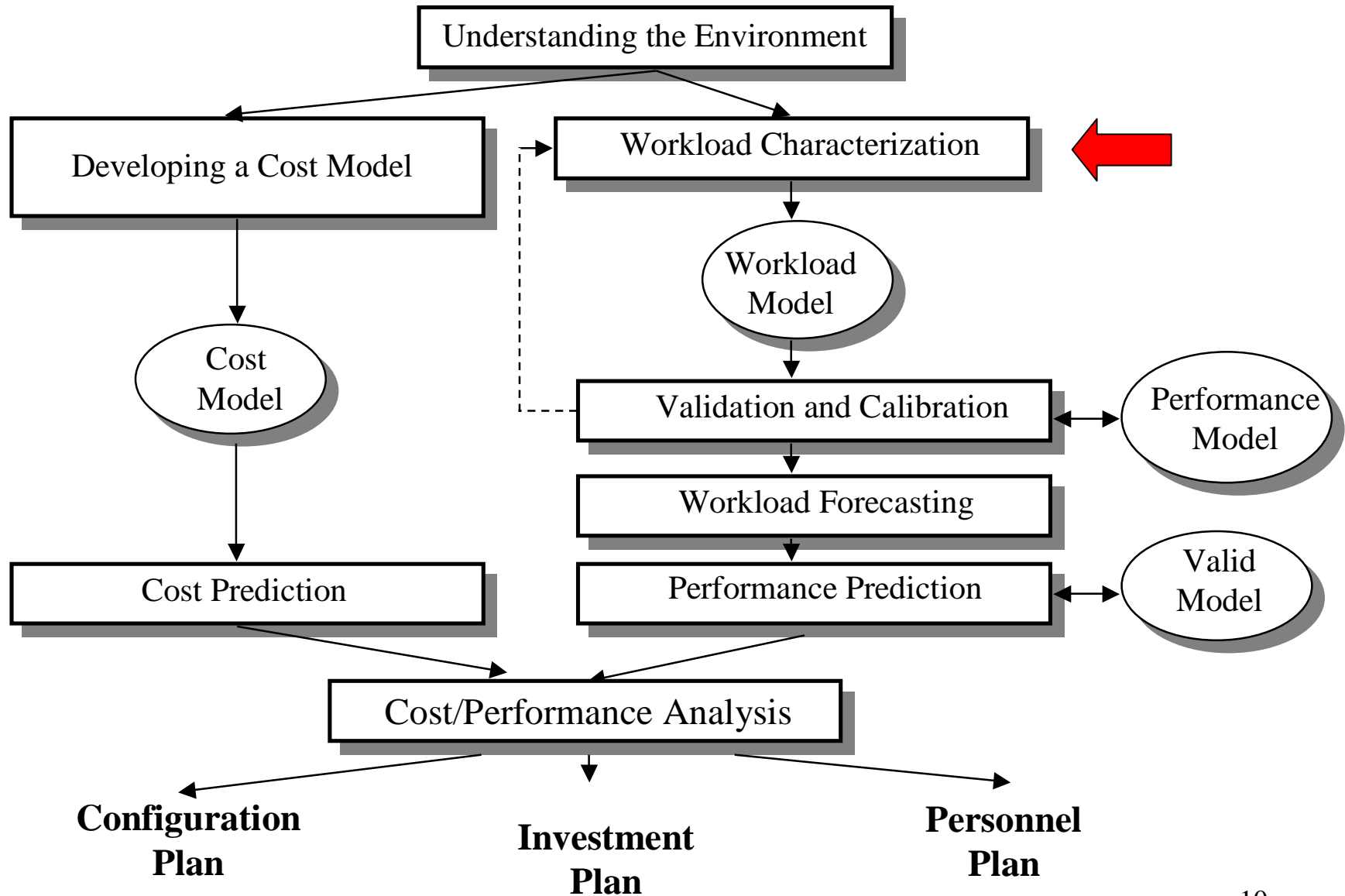
# Understanding the Environment: Hour of Day and Day of Week



day of the week: 0 is Monday  
and 6 is Sunday

Peak period: weekdays from 9 AM to 5PM.

# Methodology

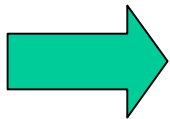


# Workload Characterization

- Basic component: HTTP gets (99.72% of all activity)
- Workload partitioning criteria: file size.
- Clustering analysis: Minimal Spanning Tree algorithm
- Out of 150 clusters, top 4 account for 98% requests.
- Zero-length transfers: 45.76% of all requests.

# Workload Characterization

- Zero-length transfers caused by:
  - version in client cache is up to date
  - requested file not found
  - access violation
- Zero-length transfers use CPU resources and generate I/O.



must take them into account!

# Workload Characterization

Class	Range (Bytes)	% Requests	% Data	Derived from
1	0	45.76	0.00	From cluster 0
2	1 - 4372	43.47	23.09	Remainder of cluster 0
3	4373 - 19284	8.78	35.41	Clusters 1, 2, and 3
4	> 19284	1.99	41.49	Remainder of clusters

- 10.77% of requests (classes 3 and 4) are responsible for retrieving 77% of the data.

# Workload Characterization: Tools used for Data Collection

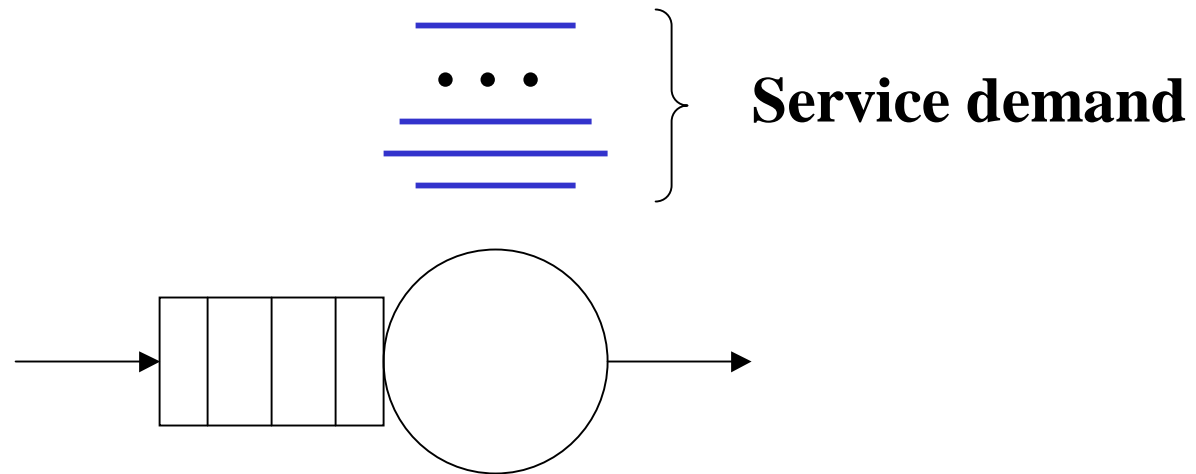
- OS:
  - iostat: disk and CPU utilizations.
  - Process accounting: detailed look at system activity.
- Web server:
  - server access log augmented through instrumentation of Apache:  
PID, service time, CPU time, and system date/time.

# Additional fields in the server access log

- PID: process id of the slave process handling the request.
- Service time: server side response time
- CPU time: accumulated CPU time of the request in msec (useless due to coarse granularity of CPU time recording by OS: 10 msec in our case).
- System date/time: system internal clock in sec.

# Service Demands

- $D_{i,r}$ : service demand of request of class  $r$  at device  $i$ . Represents total time spent by a request of class  $r$  receiving service from device  $i$ .



# Computing Service Demands

- $D_{i,r}$ : service demand of request of class  $r$  at device  $i$ . Represents total time spent by a request of class  $r$  receiving service from device  $i$ .

$$D_{i,r} = \frac{U_{i,r} \times T}{C_r}$$

← measurement interval

↑  
number of class  $r$  request completed

# Computing Service Demands

- $U_{i,r}$  is hard to measure on a production server. Need to resort to proportionality rule.

# Computing Service Demands

- $U_{i,r}$  is hard to measure on a production server. Need to resort to proportionality rule.

$$D_{i,r} = U_i \times \frac{f_r \times T}{C_r}$$

total device utilization  $\nearrow$   $U_i$

proportionality factor  $\swarrow$   $f_r$

measurement interval  $\longleftarrow$   $T$

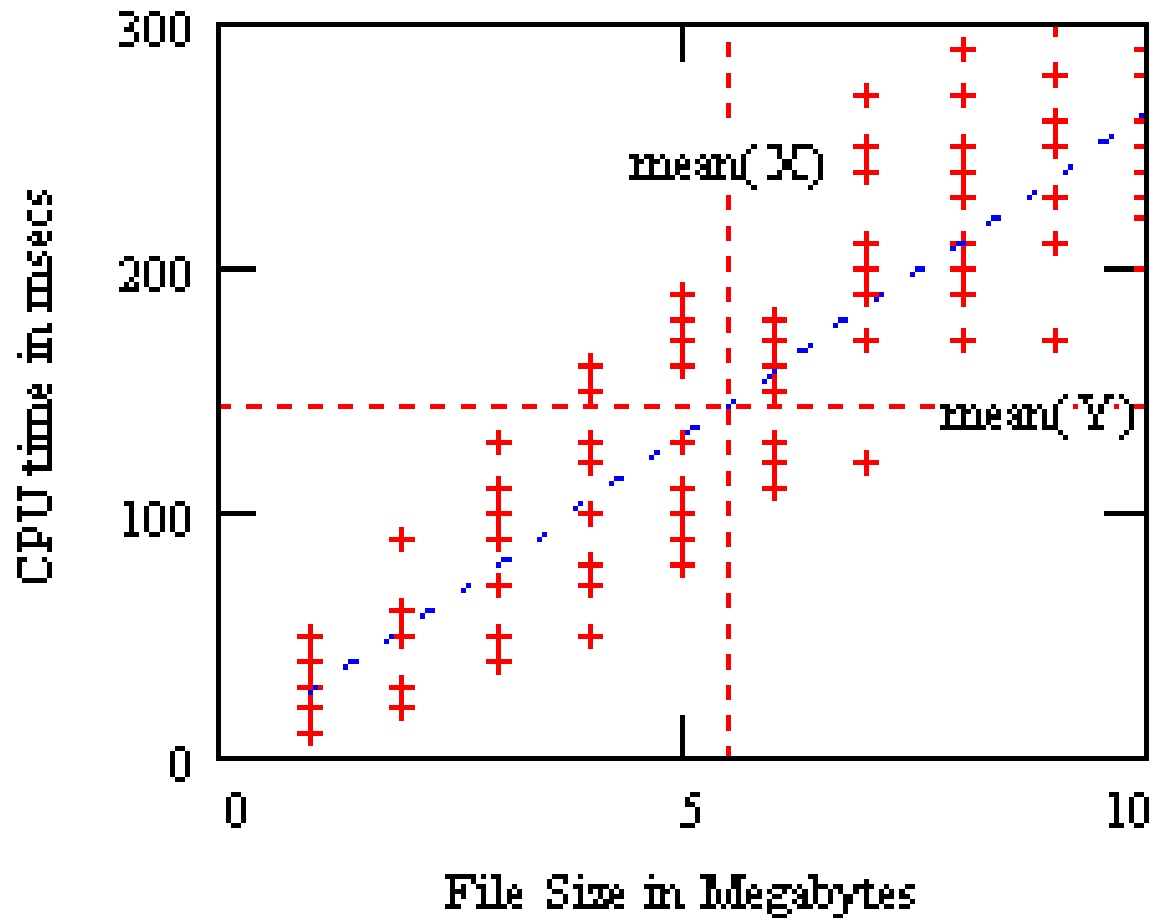
$C_r$

$\nwarrow$  number of class  $r$  request completed

# CPU Service Demands

- Conjecture: CPU service time is proportional to the size of the file retrieved.
- Developed a benchmark with files ranging from 1 MB to 10 MB in increments of 1 MB.
- Linear regression on results of benchmark.
- Y-intercept is set to be the service demand of zero-length requests (class 1).

# CPU Service Demands results of benchmark



# CPU Service Demands

$$D_{cpu,r} = \frac{\left( 2 \times U_{cpu} \times T - D_{cpu,1} \times C_1 \right) \times b_r}{C_r}$$

# of CPUs  $\rightarrow$   $2 \times U_{cpu} \times T$   
 total CPU time  $\rightarrow$   $2 \times U_{cpu} \times T$   
 CPU time of class 1  $\rightarrow$   $D_{cpu,1} \times C_1$   
 % of bytes for class  $r$   $\rightarrow$   $b_r$   
 $C_r$   $\rightarrow$  number of class  $r$  files retrieved

# Service Demands for Disks 1 and 2

- Disk 1: OS
- Disk 2: HTTP log
- Assumption: service demand for disks 1 and 2 does not depend on file size.

$$D_{disk1,r} = \frac{U_{disk1} \times T}{\sum_{r=1}^4 C_r} \quad \forall r$$

# Service demand for Disk 3

- Disk 3: document tree.
- For classes 2 to 4 use the same approach as in the CPU case.
- Need to compute the zero-length class service demand at disk 3.

# Service Demand for Disk 3

- Service demand for zero-length file class:
  - developed benchmark that accessed the I-node of files over the entire file system while the utilization of disk 3 was being monitored with `iostat`
  - `ls -lR` over 3 file systems and averaged the results

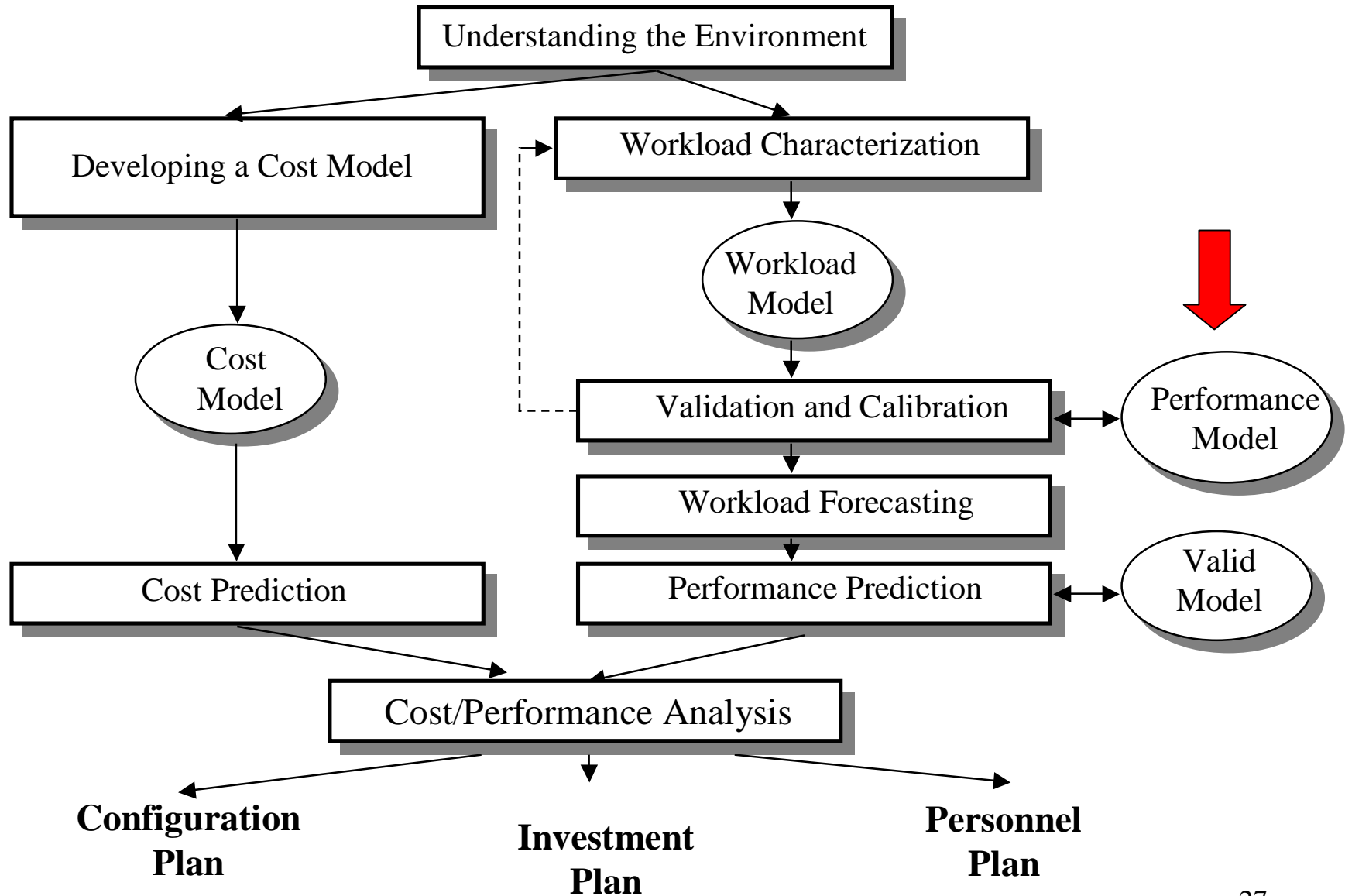
$$D_{disk3, 1} = (U_{disk3} * T) / C_1$$

# Resulting Service Demands (in msec)

	1	2	3	4
CPU 1	0.400	8.520	64.760	334.400
CPU 2	0.400	8.520	64.760	334.400
Disk1	0.740	0.740	0.740	0.740
Disk2	0.670	0.670	0.670	0.670
Disk3	1.740	1.840	13.990	72.250
In-link	0.036	0.036	0.036	0.036
Out-link	0.000	0.409	2.103	5.333

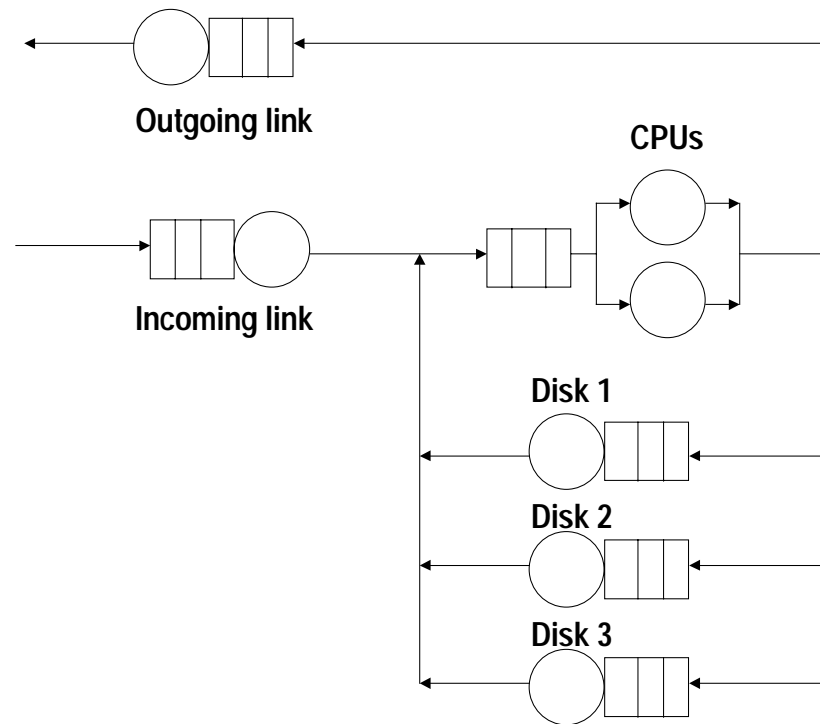
- demands for outgoing link derived from file size + protocol overhead and link capacity.

# Methodology

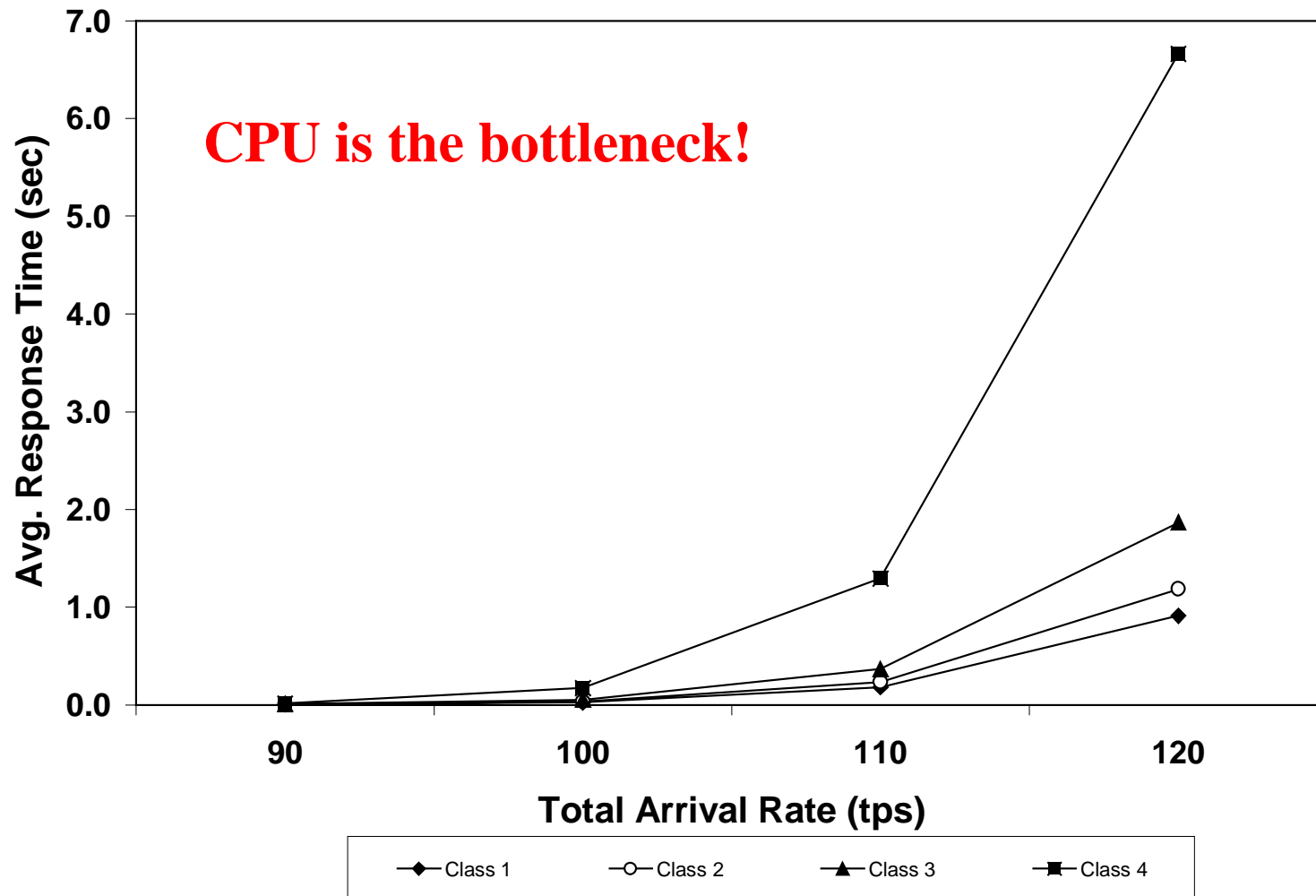


# Performance Model

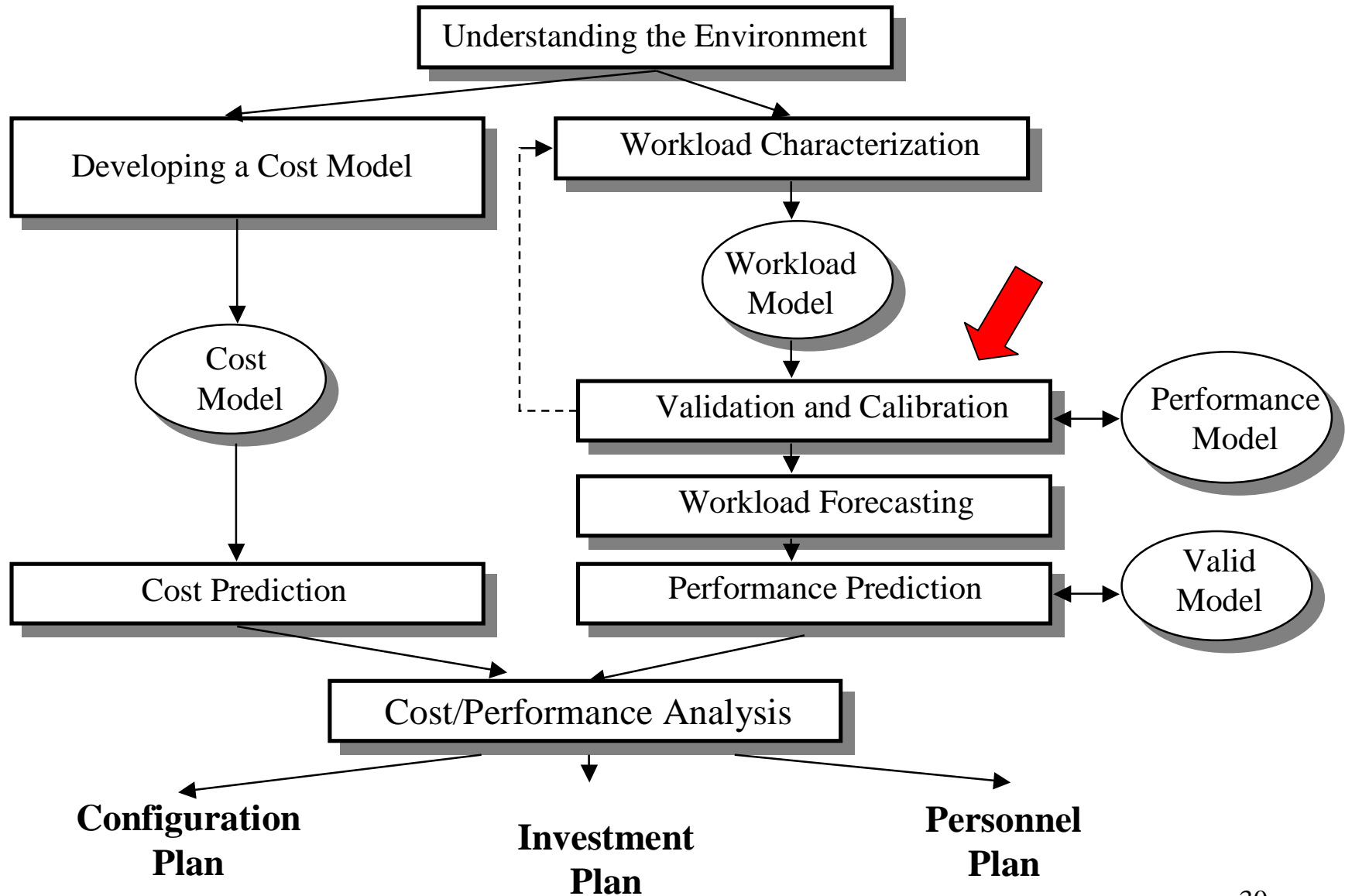
- Open multiclass queuing network model



# Performance Model Results



# Methodology



# Performance Model Validation and Calibration

- First response time comparison seemed wildly inaccurate.
- Closer look at server operation uncovered a cron job that ran every hour to gather usage statistics by reading the entire access log file.
- By the end of the month, the access log was 1.5 GB long and the cron job took 45 minutes each hour!

# Performance Model Validation and Calibration

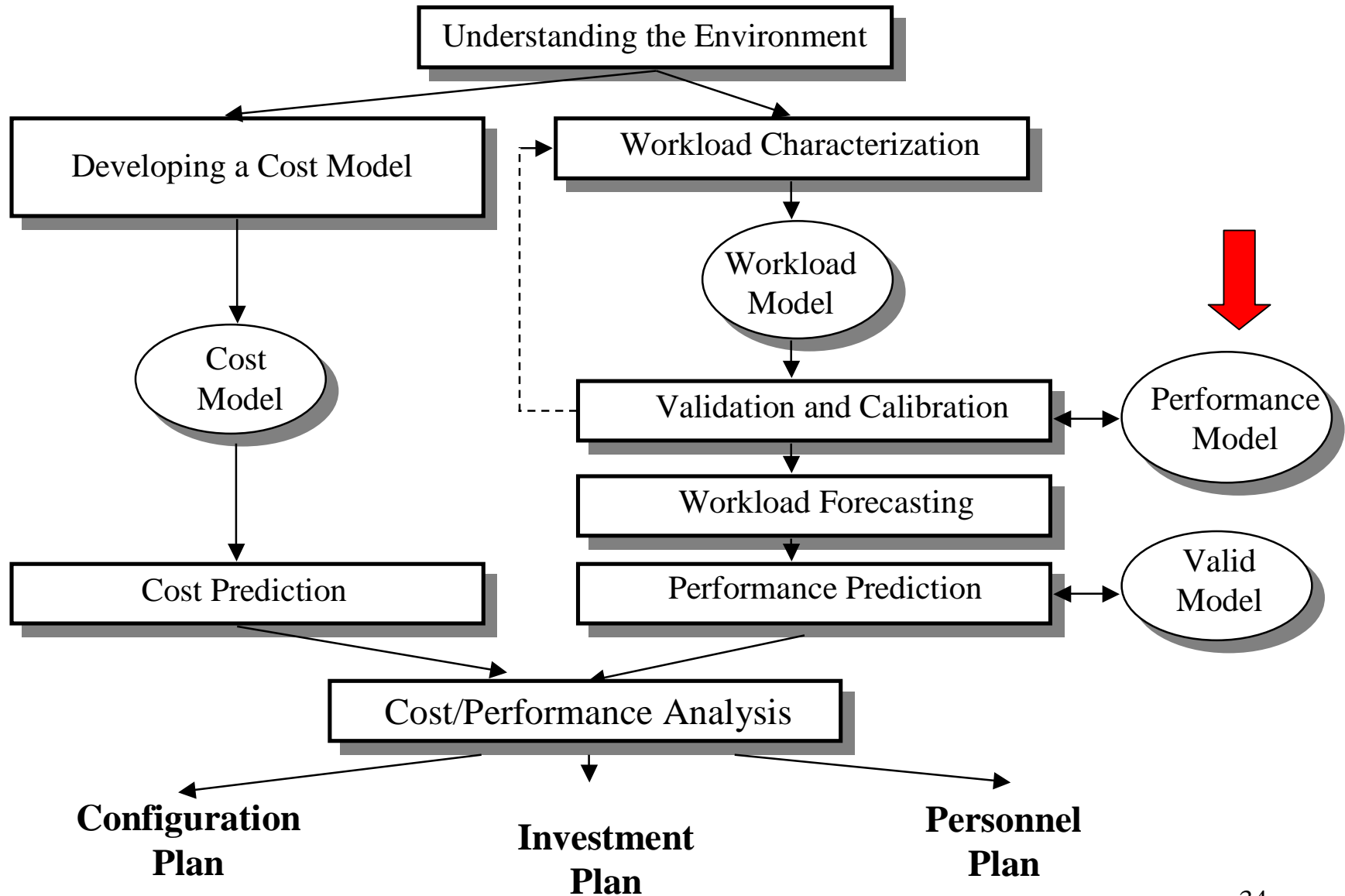
- The cron job was scheduled to run only once a week during off-peak periods.
  - Maximum percent absolute error in utilization dropped to 6.5%.

# Performance Model

## Validation and Calibration

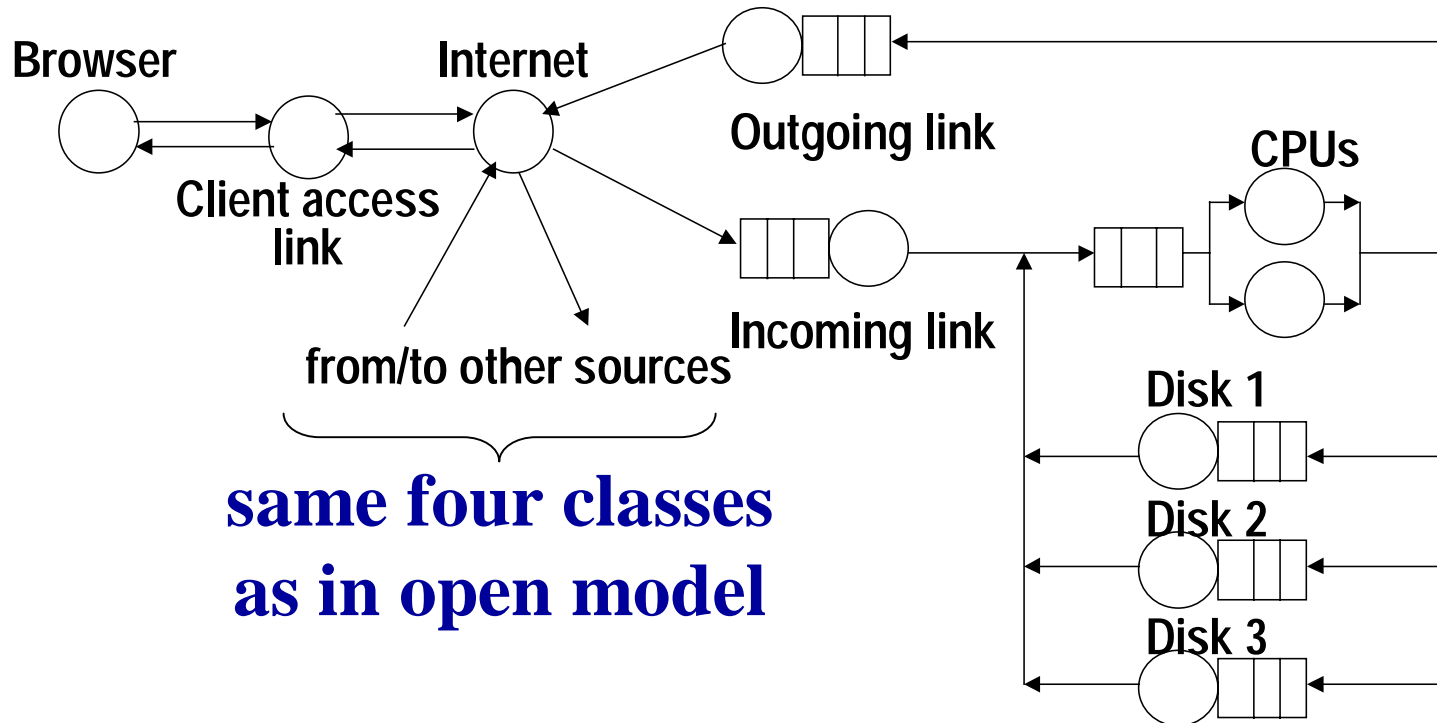
- The cron job was scheduled to run only once a week during off-peak periods.
  - Maximum percent absolute error in utilization dropped to 6.5%.
- Response time varied widely from 0.5 sec to 60 sec for the same file size.
  - Causes: network slowness, timeouts, and slow modem speeds at the client.
  - Response time is only recorded in the HTTP log when the entire file has been sent.

# Methodology

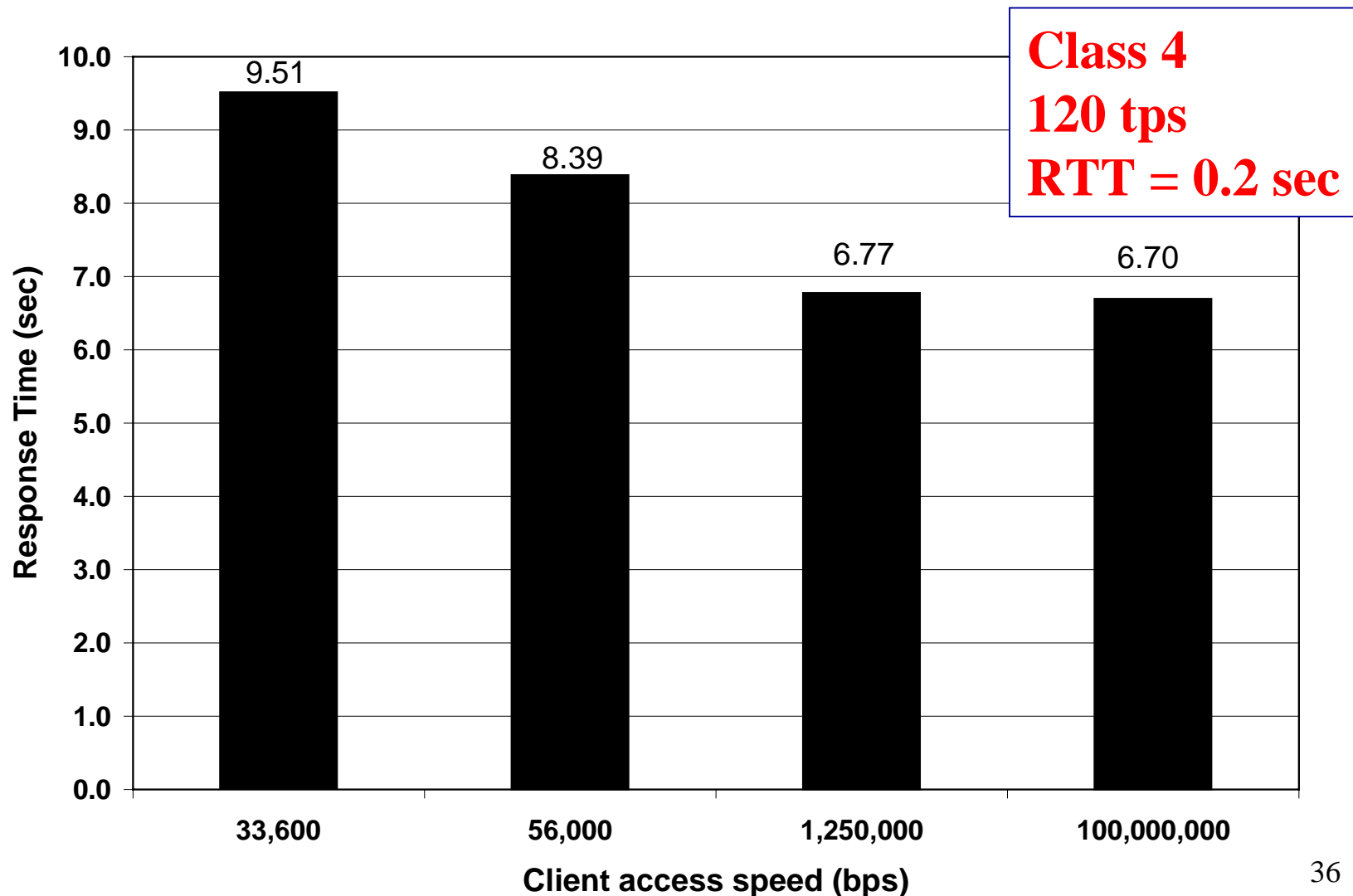


# Performance Model that Incorporates Client Access Speed

- Multiclass mixed-class queuing network model



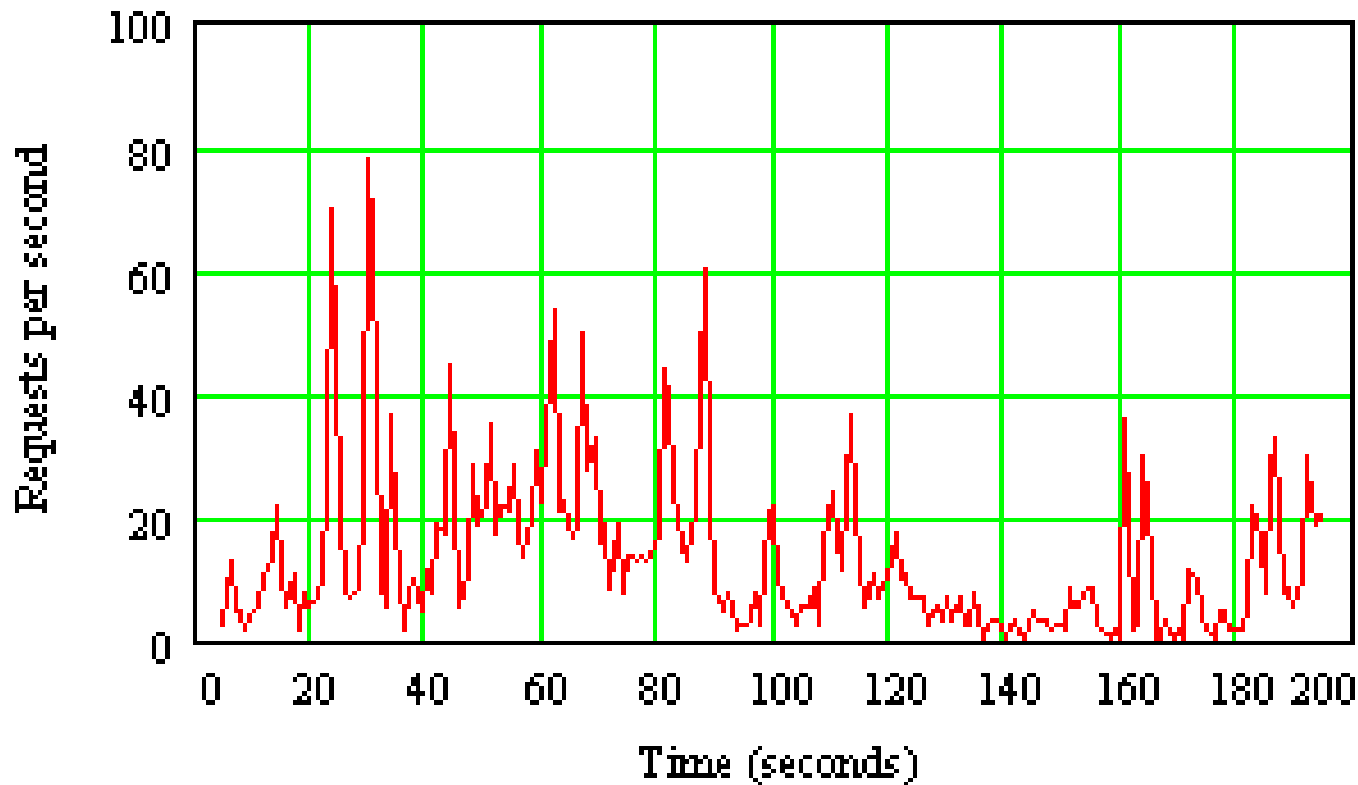
# Results of Performance Model that Incorporates Client Access Speed



# Verifying Maximum Throughput

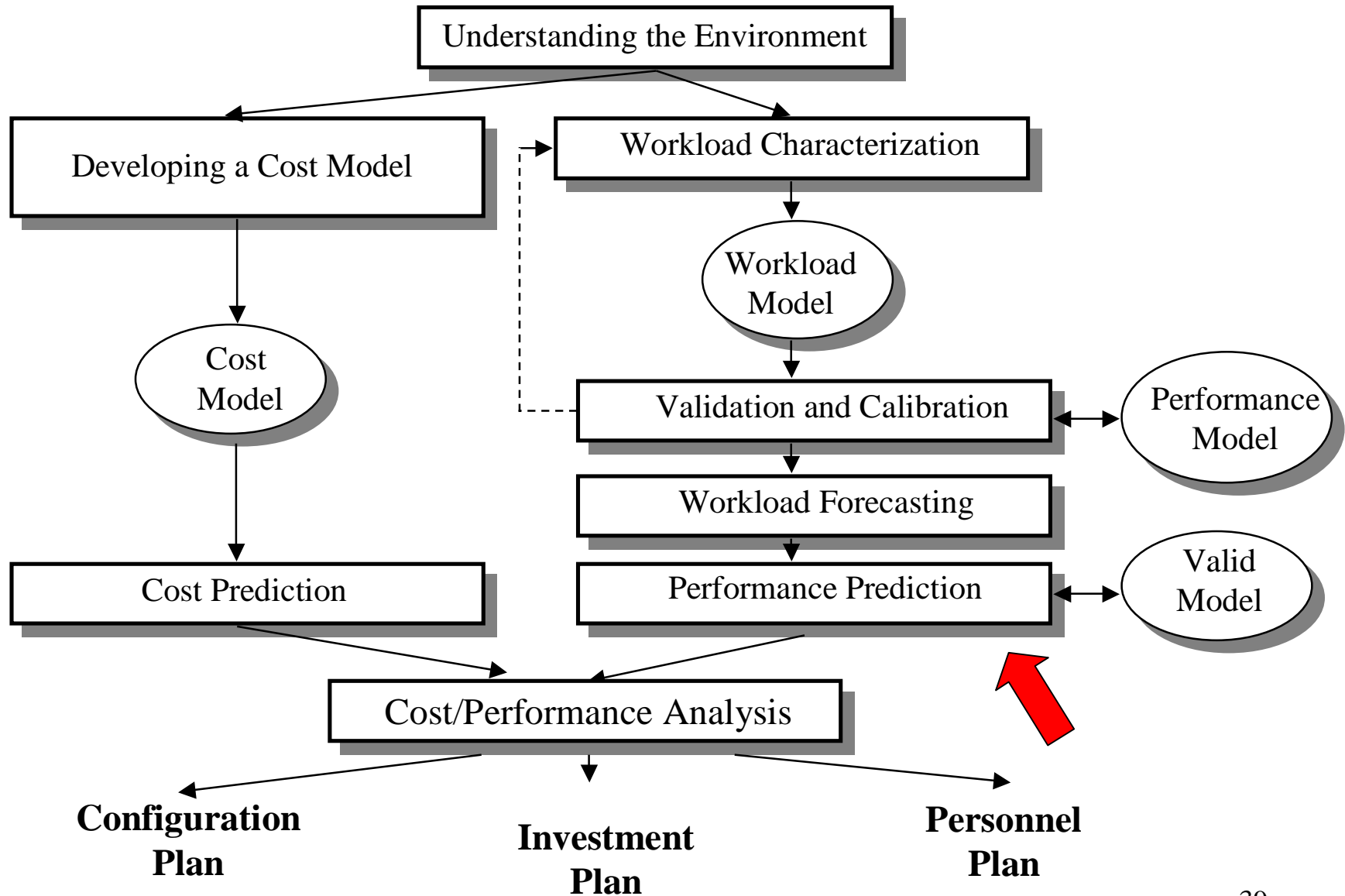
- Used InetLoad benchmark tool to drive server to maximum capacity.
- InetLoad script was derived by processing the HTTP log to preserve the bursty characteristics of the arrival process.

# Burtsy Nature of the Workload



- maximum measured throughput was 131 requests/sec, 9.2% above the maximum throughput of 120 request/sec predicted by the model.

# Methodology



# Answering What-if Questions

- The university was considering moving the Web server to a machine with RAID disks and two slightly slower processors.
- Our recommendation: use faster processors (250 MHz as opposed to the 168 MHz ones).
- Model predictions: response time for class 4 went down to 0.592 sec from 6.6 sec and the maximum throughput increased 52%.

# Concluding Remarks

- Major challenges in conducting a capacity planning study are:
  - devising a sound data collection procedure to obtain parameters for the performance model
  - validating the model
- This study showed importance of client access speeds in modeling server side response time.

# Bibliography

- Capacity Planning for Web Performance: metrics, models, and methods, D. Menascé and V. Almeida, Prentice Hall, 1998.
- Capacity Planning and Performance Modeling: from mainframes to client-server systems, D. Menascé, V. Almeida, and L. Dowdy, Prentice Hall, 1994.