

Multitiered Architectures

CSE 794R/ECE 694R

1

So Far...

- We've been looking at "flat" applications
- Very simple C/S model
 - temperature converter
 - compute engine
- Protocol: RMI

CSE 794R/ECE 694R

2

Applications

- Large class of apps involve UI & data
- Logical structure
 - UI --> graphical events (mouse movement, button pushing)
 - Application --> processing, calculating
 - Data --> persistence, triggers
- Databases
 - relational (tables): traditional business apps
 - object-oriented: science & eng apps

CSE 794R/ECE 694R

3

Example: Web Search Engine

UI		} UI level
(keyword exp)	(html page)	\
Query	HTML Generator	\
Generator	(ranked list)	> App level
	Ranking component	/
(database	(result: titles	/
query)	and metadata)	/
	Database	} Data level

CSE 794R/ECE 694R

4

2-Tier Architecture

- Chop this into client & server
- Lots of choices where to divide

UI <- a

<- b

Application <- c

<- d

Data <- e

CSE 794R/ECE 694R

5

2-Tier Architecture cont.

- a. "ultra thin" client
 - app has complete control over data present
 - "dumb terminal" (mainframes, just echo keystrokes)
- b. UI on client
 - protocol with app could be app-specific
 - no processing (other than what needed for display)
 - e.g., browser that turns html into displayed image
- c. UI + some app
 - e.g.: a form to be filled out
 - checked at client side for errors
 - reduces network congestion

CSE 794R/ECE 694R

6

2-Tier Architecture cont.

d. main variant of "2-tier"

- client connects directly to database manager
- functionality and logic all at client side
- "thick client" (or "fat client")

e. client stores some data

- e.g., maintain a cache of freq stuff

CSE 794R/ECE 694R

7

Shortcomings

- In Java, access database through JDBC (Java Data Base Connectivity) + proprietary drivers
- In 2-tier architecture app:
 - how do you maintain/modify application logic?
 - how do you upgrade database drivers?
- Require changes to (all) clients

CSE 794R/ECE 694R

8

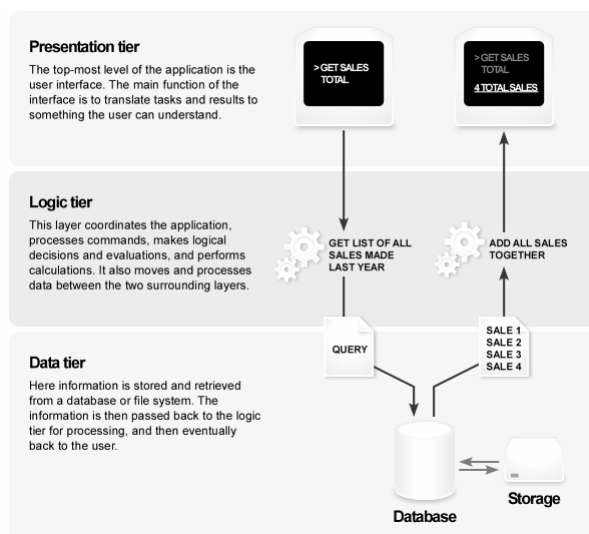
3-Tier Architecture

- Keep these 3 levels logically separate (and typically *physically* separate too)

client
(http or rmi/corba)
application
(jdbc)
data

CSE 794R/ECE 694R

9



CSE 794R/ECE 694R

10

3-Tier Architecture Gains

- Performance
- Flexibility
- Robustness
- Security
- Scalability

4-Tier Architecture

- Further separation of application level
 - client (java, applet, javascript)
 - presentation layer (servlet, JSP)
 - business logic (EJB)
 - data (JDBC)

N-Tier Architecture

- Further separation of layers
 - client
 - caching (clients at same wan can share downloads)
 - presentation
 - business logic
 - data access
 - data

Vertical vs. Horizontal Distribution

- Horizontal: multiple servers at the same level
 - fault-tolerance
 - increased throughput
 - (load balancing gives good (?) predictable (?) performance)
 - incremental deployment of modifications
- Vertical: adding resources at different levels
 - e.g., cpu's, memory, storage
- ("Diagonal" scaling: a combination of the two)