E-COMMERCE and SECURITY - 1DL350

Spring 2013

An introductory course on e-commerce systems

alt. http://www.it.uu.se/edu/course/homepage/ehandelproject/vt13/

Kjell Orsborn
Uppsala Database Laboratory
Department of Information Technology, Uppsala University,
Uppsala, Sweden

Web applications, tools & architectures

ch 6, 7

Kjell Orsborn

Department of Information Technology
Uppsala University, Uppsala, Sweden

UPPSALA
UNIVERSITET

The Internet

- The Internet is an open system
  - Details publicly available
  - A lot of software is free
  - Lots of publicly available expertise available via such things as newsgroups
  - Dangers with privacy
- Implications of open systems
  - Wide variety of implementations, for example of TCP/IP
  - Cost of implementation less
  - High level of compatibility
  - Wide variety of developers selling products
- Examples of open systems and code
  - HTTP, TCP/IP, Java, Linux, Apache
- The Internet has a layered architecture
  - Level of functionality, where each level draws upon facilities in a lower level
  - As you proceed downwards you get nearer the computer
  - Achieves separation of concerns

Brief history of Internet

- Internet history (i)
  - ARPA (Advanced ressearch Projects Agency, later DARPA) started the ARPAnet network (1969)
  - ARPAnet originally used the NCP protocol
  - 1974 Cerf and Kahn developed TCP/IP
- Internet history (ii)
  - Splitting of ARPAnet into MILnet and ARPAnet (1983)
  - The term Internet (introduced 1974) came into more general use in early 80’s.
  - The World Wide Web at CERN was created in 1989 by Sir Tim Berners-Lee
  - Development of new protocols to cope with huge growth

Internet protocols

- *Telnet*, used for connections
- *File Transfer Prototcol* (FTP)
  - used for file transfer
- *Simple Mail Transfer Protocol* (SMTP)
  - used for electronic mail
- *Kerberos*
  - used for security functions
- *Network File System* (NFS)
  - used for transparent file sharing
- *Trivial File Transfer Protocol* (TFTP)
  - used for fast transmission of files
- *Transmission Control Protocol* (TCP)
  - used for fast transmission of files.
- *User Datagram Protocol* (UDP)
  - used for fast transfer of data, unreliable.
- *HyperText Transfer Protocol* (HTTP)
  - used for transferring Web documents
- *Internet Protocol* (IP)
  - basic functioning of moving data

Client and servers

- A network can be envisioned as a set of clients and servers
- Servers provide a service, for example a Web server delivers Web documents or dispensing files.
- Clients call on the services provided by a server
- The distinction is not hard and a server may act as a client to another server.
- A server acting as a client:
  - In an ecommerce application, a Web server might call on the service of a database server in order to access some data such as catalogue records

Some servers

- File servers
- Database servers
- Groupware servers
- Web servers
- Mail servers
- Object servers
- Print servers

Web servers

- In e-commerce terms, the most important type of server
- Deal with in detail later
- Stores HTML files and dispenses them to clients
- Processes forms details
- Communicates with other servers, for example database servers

Database servers

- Next to web servers the most important type of server for ecommerce
- Explained in more detail later
- Stores relational databases
- Responds to queries in language called SQL

Tiered architecture terminology

- Distributed architecture
  - System composed of programs running on multiple hosts
- Tier
  - One of those host computers
  - But…can have virtual distributed apps running on a single host
  - Tier can also signify a logical partition of processing
- Examples:
  - Client
    - e.g. web browser
  - Server
    - Object server
    - Enterprise server
    - Database server
    - Web server

… more terminology

- Presentation logic
  - How information is presented to the client
- Business logic
  - Collection of objects and methods which are different from business to business, e.g. flight, customer, checkAvailability(), …
- Data logic
  - How to ensure data is persisted, secure, and transactionally safe

Tiered architectures

- An example of separation of concerns
- Most popular model has three layers
- Developed for maintenance reasons
- Also have important security implications
- Importance of tiers
  - allow separation of concerns
  - coding paradigms different for each tier
  - required skill set differs too

Along with security, this is probably the most important aspect of e-commerce system design

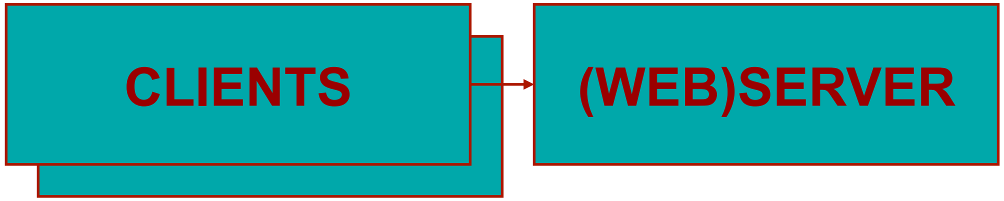
1 tier

STANDALONE APPLICATION

+ Simplicity – no networking
+ High-performance
+ Self-contained

- Can’t access remote services
- Potential for spaghetti code

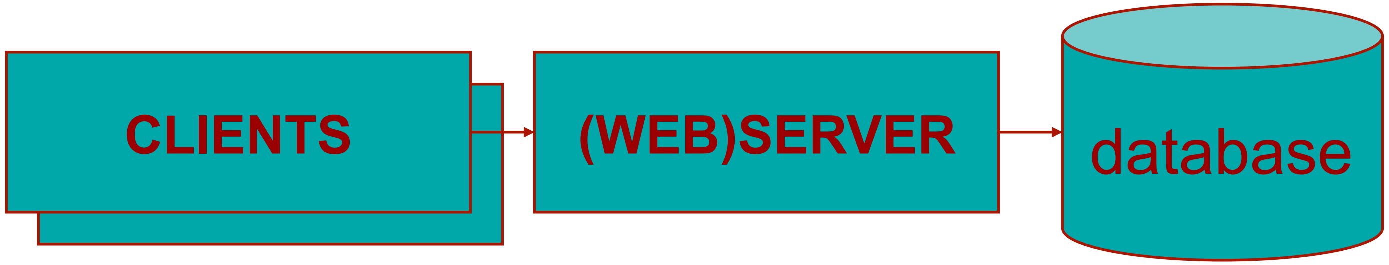
2 tiers



- + Quite simple
- + Separation of presentation logic from business logic

- - Little potential for resource sharing, a big problem for ecommerce applications

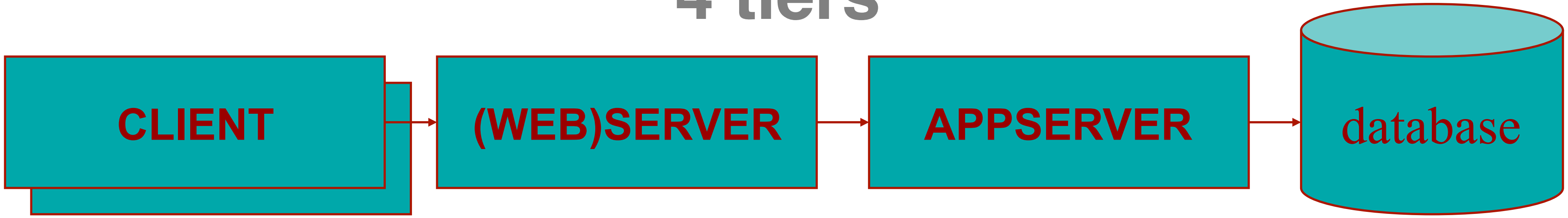
3 tiers



- + Separation of presentation, business and data logic
- + Concurrent data access
- + Shared resources

- - More expertise required
- - More security
- - might need object-relational mapping

4 tiers



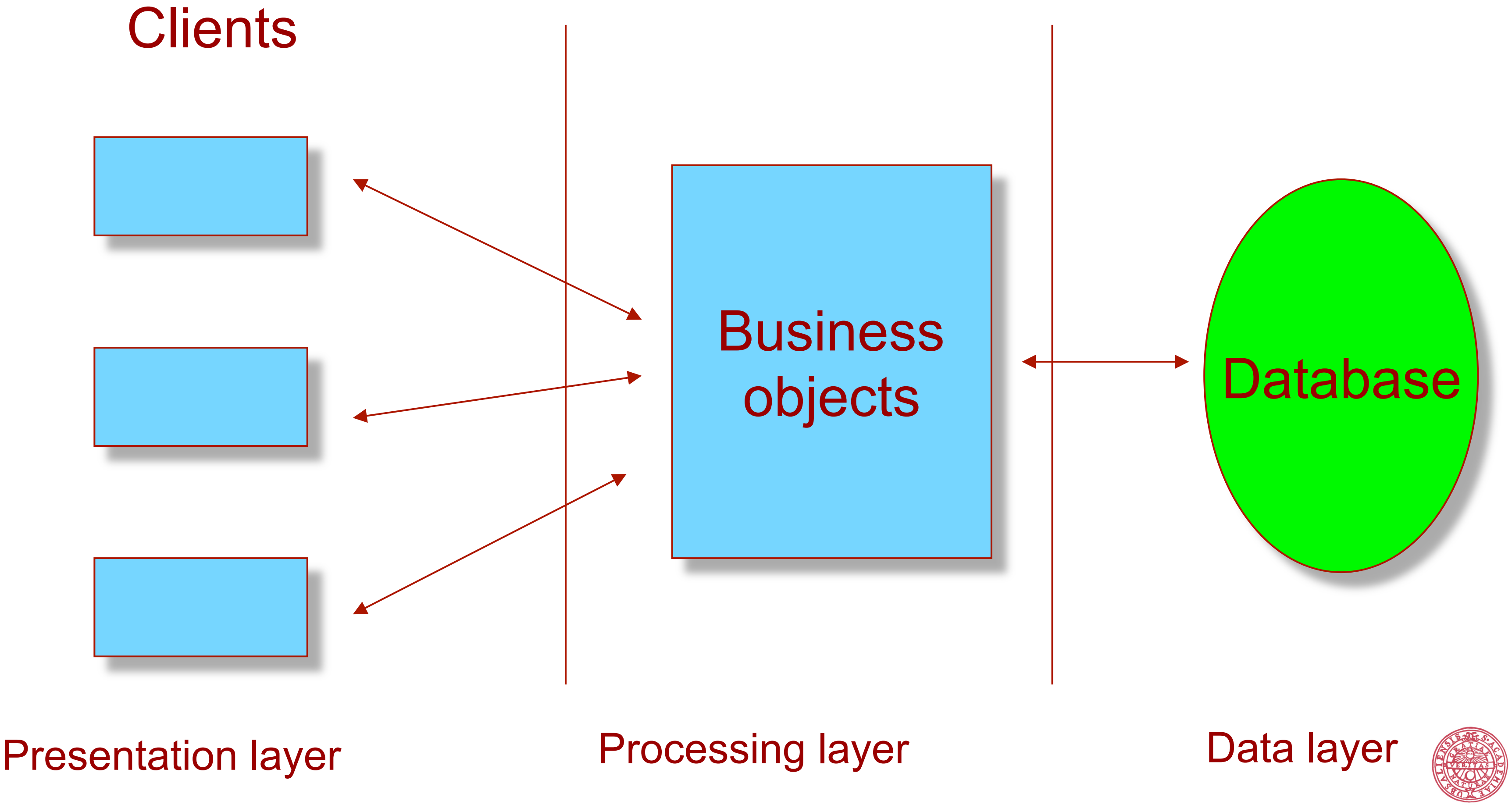
- + (near) automatic handling of transactions, security, persistence, …
- + supports just about anything

- - learning curve
- - can be inefficient due to generality
- - expensive (but see JBoss)

Problems with tier classifications

- HTML form communicating with a web server
  - 1.5 tier systems (is web form a program?)
- Applet running on a browser, downloaded from web server
  - 1 tier, but depends what the applet does

Another look at the Three-tier model



Three-tier model

- Presentation layer contains HCI for client
- Processing layer contains business objects
- Data layer contains stored data
- Rationale:
  - HCI can go on the client and does not require to be transmitted over network
  - Business objects reflect domain entities in application, for example in a sales site: catalogue, product and customer
  - Business objects shield the implementation of data
  - All application programming done on business objects
  - Details of underlying data hidden to the application programmer, for example the programmer should be unaware of the database technology

Middleware

- Software used to support interactions between clients and servers
- General middleware and service middleware
- General middleware used for application neutral functions
- Service middleware associated with a particular services such as that provided by a Web server

An example of middleware

- Queues which interpose between clients and servers
- Clients place data and transactions on the queues
- Servers remove data and transactions
- Simple model often used to interface legacy applications and implement mobile applications

Protocols

- Used for communication within a distributed system
- Used in message passing
- HTTP is the protocol used for Web server access, described later
- Many other protocols exist, for example POP3 for email
- Simplest is the request/response type of protocol
- Can be fixed, protocol does not change, for example HTTP
- Can be adaptable and negotiated, for example SSL negotiates a protocol subset
- Can be synchronous or asynchronous

Synchronous and asynchronous protocols

- Synchronous means that entities work in step with each other, for example as in a request response protocol
- Asynchronous protocols are not bound by co-ordination, good example are those associated with message-oriented middleware

Request-response protocols

- Simple type of protocol
- A client making a request receives a response
- HTTP best example
- HTTP has a command which requests an HTML file, the response is either the file or an error message

Protocols can be application specific

An example of The POP3 protocol:

`USER`	User is going to retrieve mail
`PASS`	Here is my password
`STAT`	How many emails waiting?
`DELE`	Delete an email message
`RETR`	Retrieve some messages

Why client server?

- Openness
- Scalability
- Specialisation
- Reliability
- Design flexibility

Opennesss & Scalability

Openness means that a number of different platforms can be used in a network, all that is needed is some common protocol for them to communicate

Scalability means that more and more servers can be added to a network as application demand increases. Note, though, that the increase in power will not be linear in terms of the number of servers

Specialisation & Reliability

Specialisation means that servers can be designed specifically for some service, for example acting as a mail server, with no performance compromise because they have to carry out some other service

Reliability can be achieved by duplicating programs and data around a network; this means that when one server malfunctions another takes over

Design flexibility

Design flexibility provides a greater solution space than that achievable with single computer models. For example data can be kept close to a user resulting in faster response times.