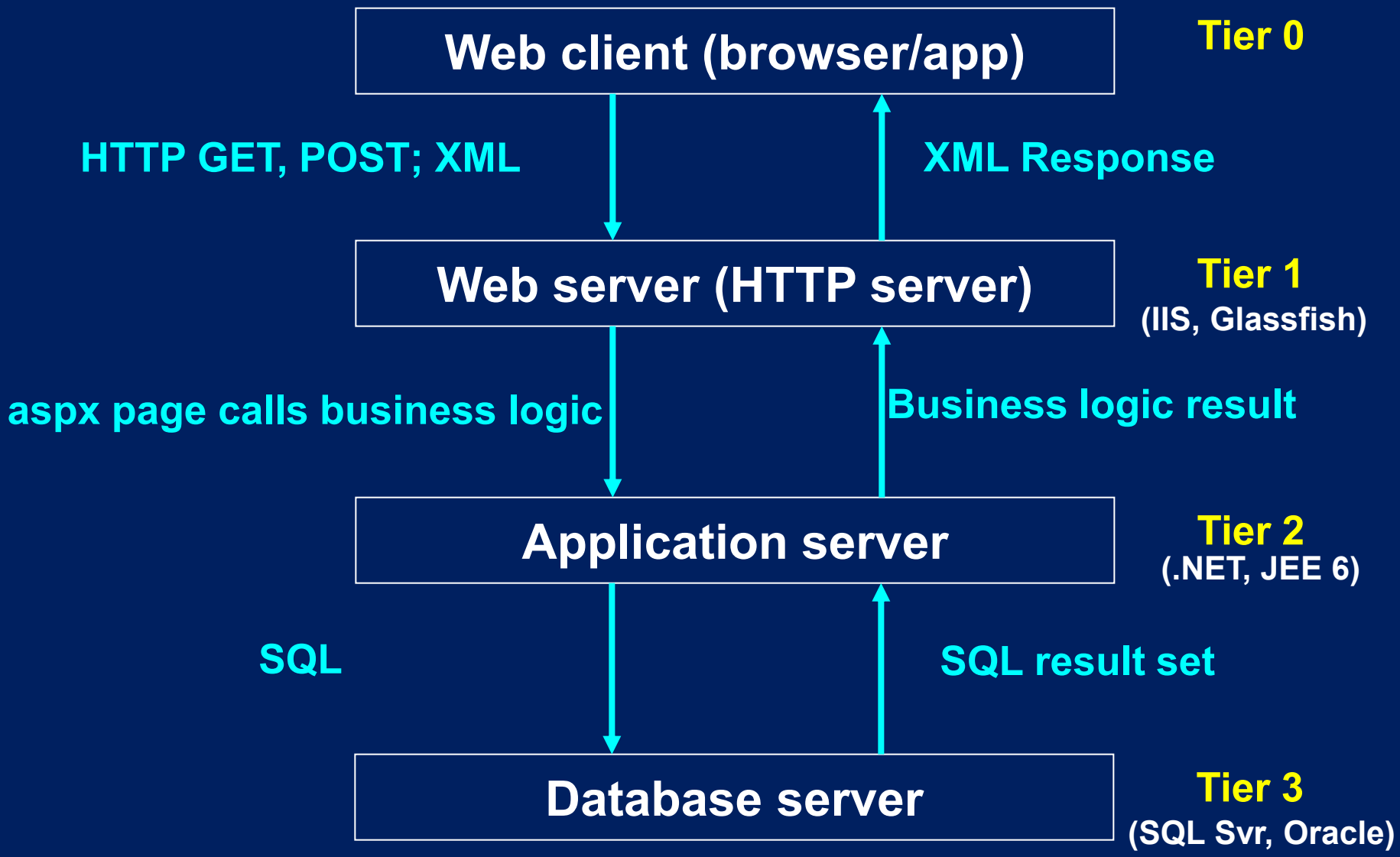
1.264 Lecture 30

System architecture

Next class: Exercise due after class
Please start Visual Paradigm-we'll use it in class today
No zip file to download-it's online

Three tier architecture



In HW7, your aspx pages include the business logic that is usually in the application server

Case study: Travel industry

- Open Travel Alliance (www.opentravel.org)
  - 150+ airlines, hotels, car rental, GDS, others
    - GDS: global distribution system (Sabre, Apollo, etc.)
  - Collaboration to implement industry-wide open e-business specifications
  - Electronic distribution supply chain
    - Developed common XML formats, trading partner to trading partner
  - Goals are to lower supply chain costs and improve consumer travel costs and experience
- Current conditions
  - Two protocols in current use
    - ResTeletype: 64 character sentence
    - UN/EDIFACT (EDI): fixed messages, difficult to change
  - Same systems, messages have been used for 30 years

"T" travel agency

- US travel agency "T" in Midwest:
  - Lines of business:
    - Airline, hotel reservations for residents
    - Local attraction reservations for international visitors
  - Systems used:
    - Telephone, fax, teletext message service
    - Internally developed computer apps: customers, local attractions, etc.
  - Web changed travel business
    - "T" became GDS partner for international travel in Midwest
    - New suite of services offered: air, hotel, rail, car rental
    - New suite of international services for US customers
    - Opened branch offices around the world

Based on Kumar, Narayan, Ng, "Implementing SOA Using Java EE"

"T" travel agency, cont.

- "T" changed its information systems to work with GDS and its consumer customers
  - Application software, middleware, desktops, servers
- "T" unable to penetrate international travel market
  - To remedy this, it bought UK and Asian travel agencies
  - All three companies had different legacy applications, and networks, which created problems
- Systems architecture needed for new company
  - Meet evolving customer and GDS requirements
  - Minimize legacy systems
  - Retain existing Web services, if possible
  - Create new Web services to replace legacy apps

Activity diagram: current system

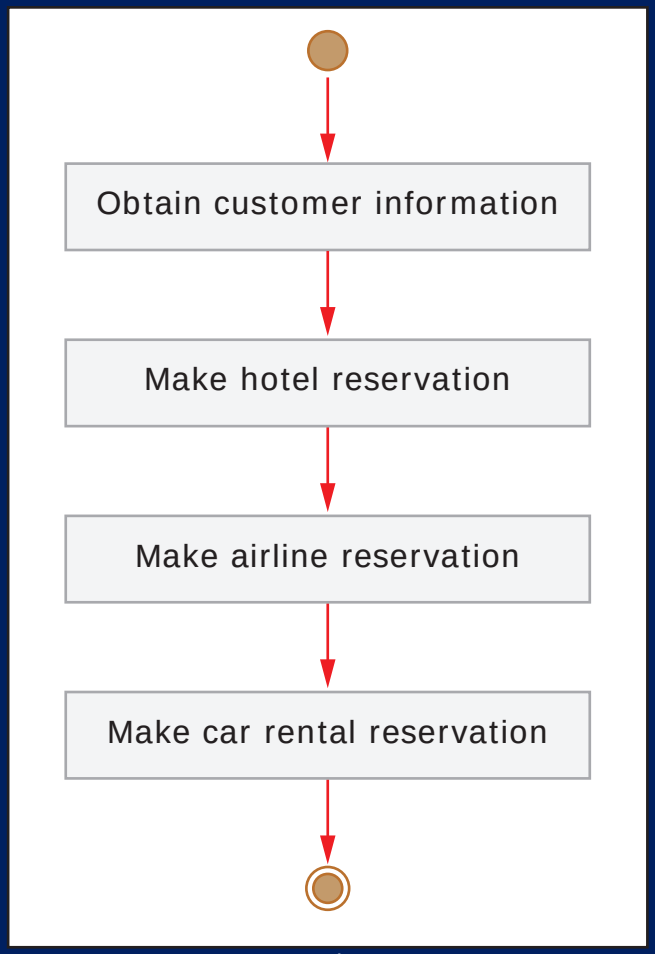


Image by MIT OpenCourseWare.

- **Problems:**
  - **Choices are complex:**
    - **Air: carrier, price, time, hops,…**
    - **Hotel: chain, room type, rate, location, …**
    - **Car, sightseeing package similar**
  - **Airline reservation may bundle hotel, car reservations at discount**
  - **Reservation confirmation may be delayed (especially international)**
  - **Partial itinerary not acceptable to customers today**
    - **Customers change their mind partway through the itinerary process**
  - **Travel packages must be flexible, attractively priced, easily booked**
    - **Customers have own discounts: AAA, …**
  - **Sequential process too slow for online users**

Exercise: Activity diagram

- Draw a UML activity diagram:
  - Do as many steps from previous diagram in parallel as possible
  - Check that parallel results fit together
  - If results fit together: present the outcome to customer
    - Otherwise, re-do required step(s) until a fit is found,
    - Or tell/ask customer to search again

Solution

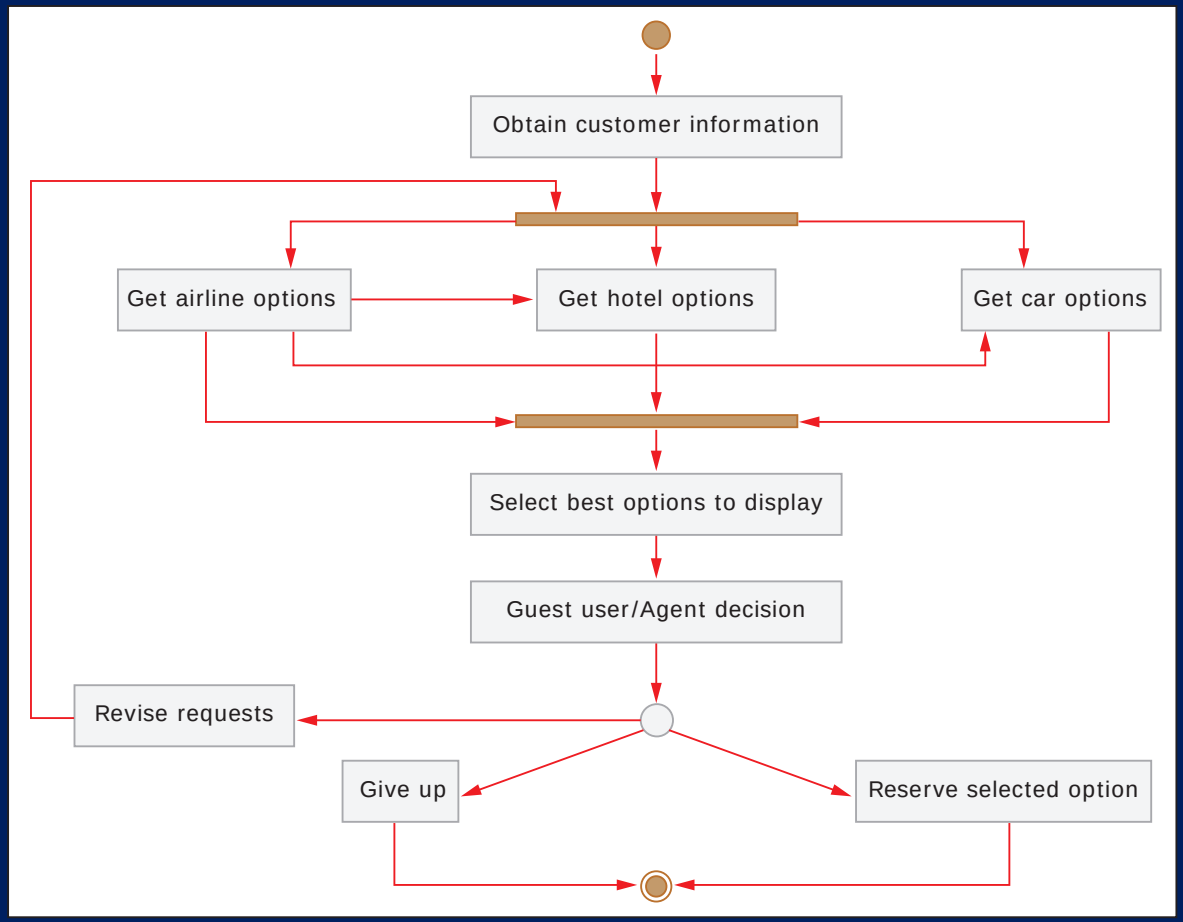


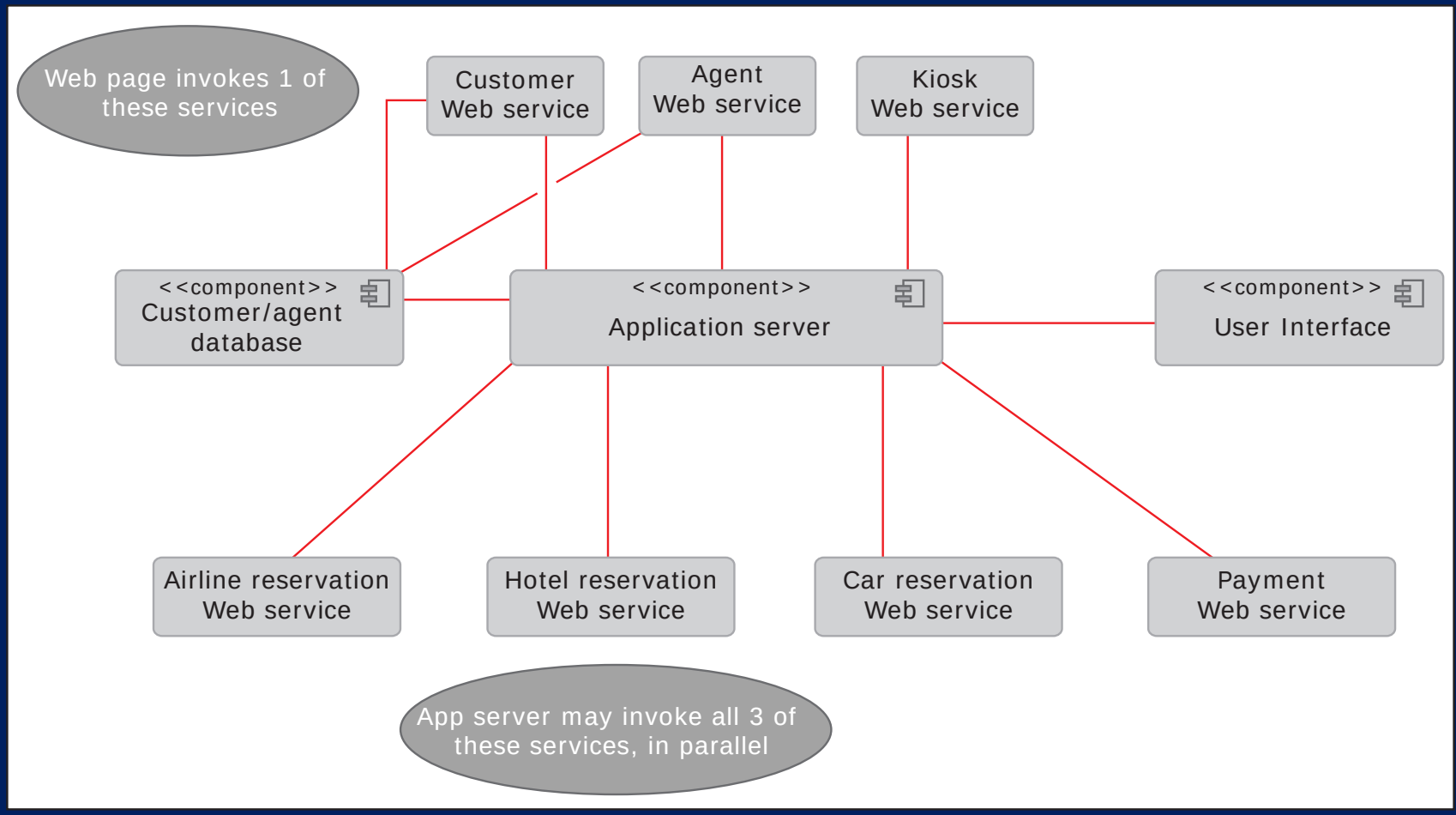
Image by MIT OpenCourseWare.

- This models a set of Web services, providing reservations for multiple customers connecting to multiple reservation systems.
- Reservation itinerary gets filed with “T”
- “T” checks if air res made; if not it uses air Web service.
- “T” then checks hotel, car in same way

Solution, cont

- This architecture will generate many (often thousands) of queries in parallel
- Most of the results won't be used directly
  - They are sorted and assembled by the application
  - Only the best ones are shown to the user
- This requires much more (often thousands of times more) bandwidth than older approaches
  - Fiber optic bandwidth is large and can support this
- Supply chain and other business applications are moving to this model
  - Query many carriers, vendors, etc. for availability, price, …

System architecture based on activity diagram



"Wrapper" legacy code: almost all allow Web services to be added. Only a few Web services needed per legacy app to interface to it.

Class diagram for application server

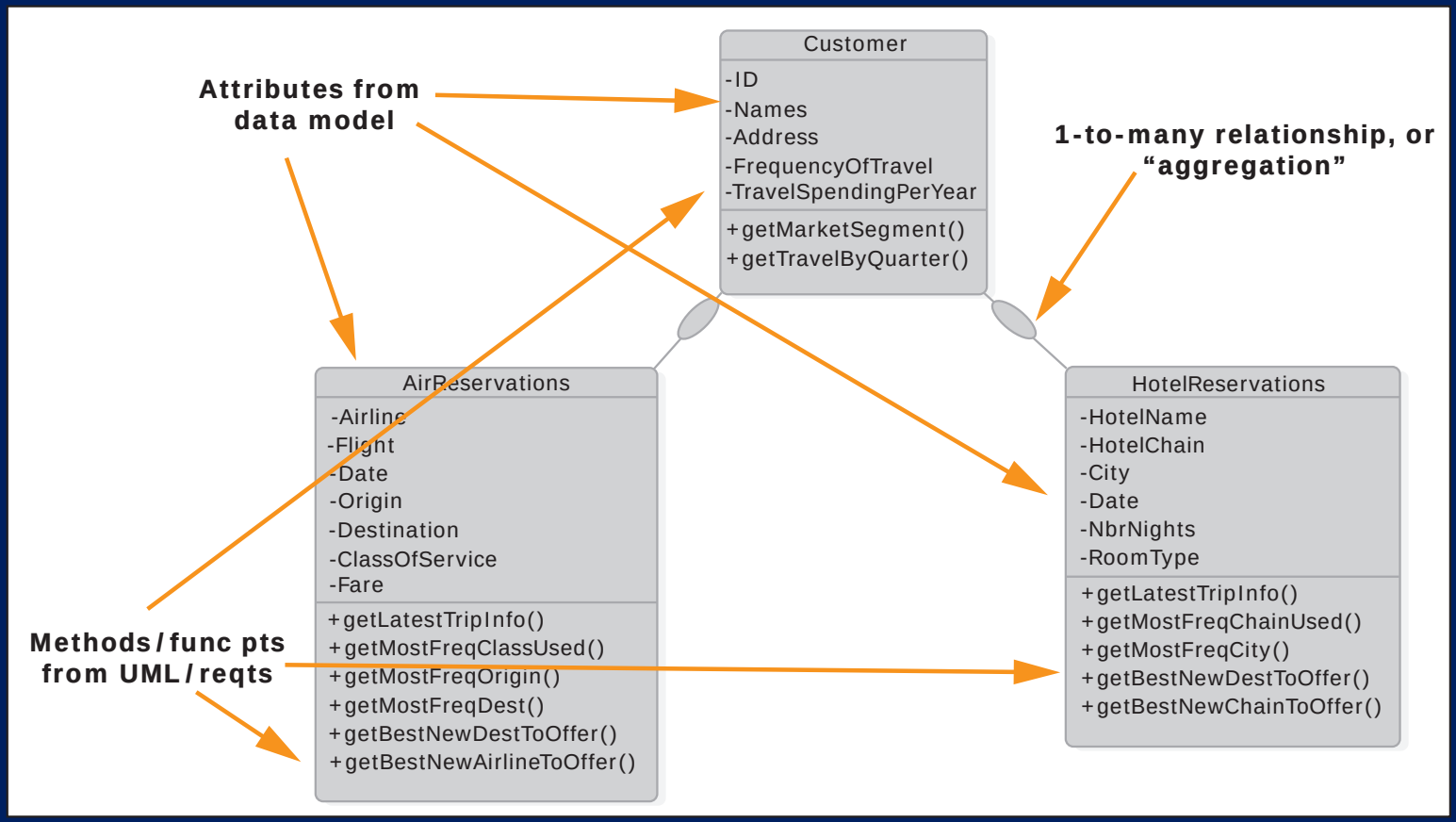


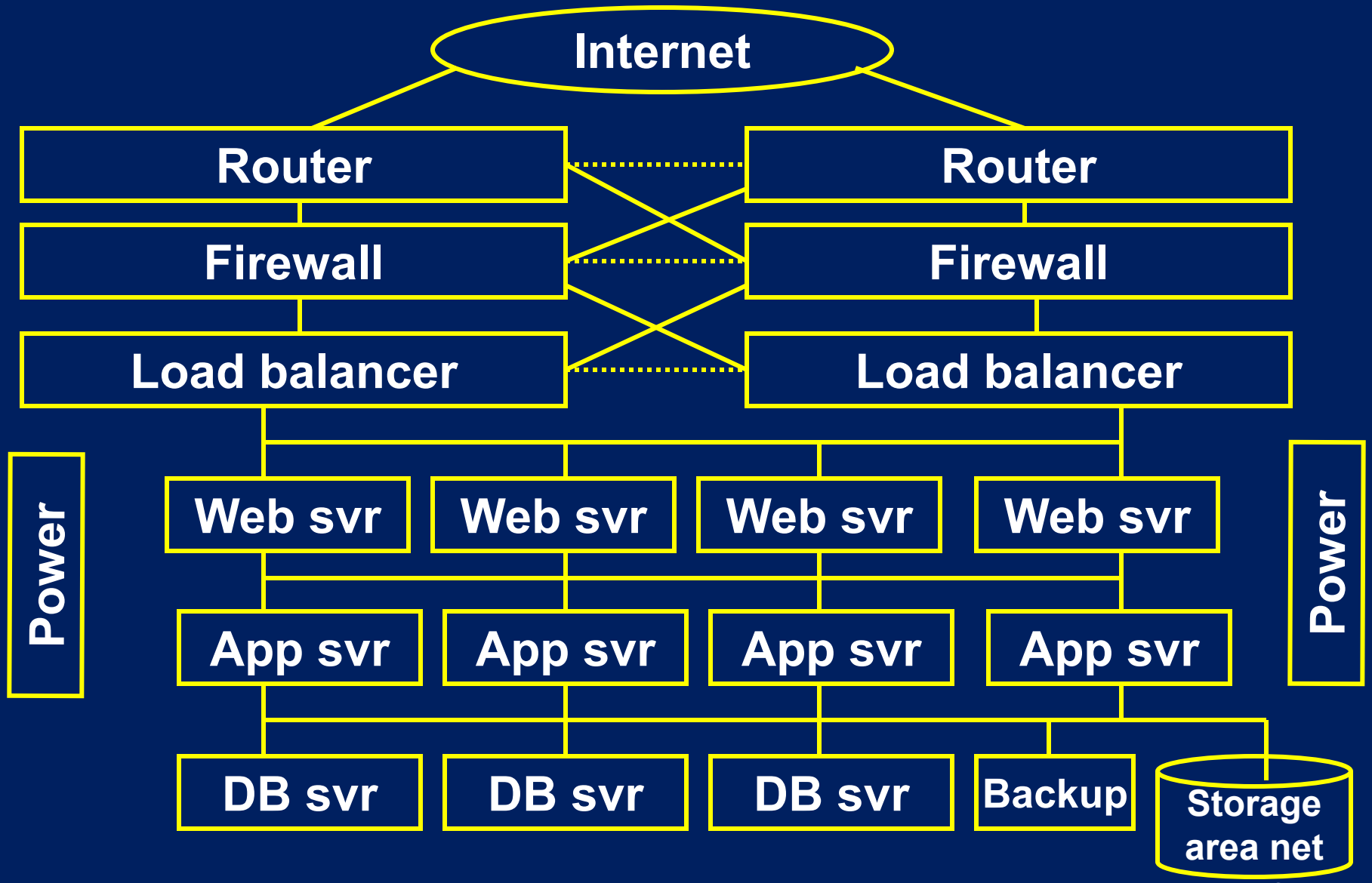
Image by MIT OpenCourseWare.

Similar classes for car rental, itinerary check, etc.
Write these methods in C#, similar to those on .aspx pages
App server and its tools handle db, xhtml, xml input/output

System architecture components

- Application components
  - Web server
  - Application server
  - Database server
- Network interface architecture components
  - Router
  - Firewall
  - Load balancer
- All include the hardware and the software that runs on it

Architecture components at server end



Simple example of system hardware architecture

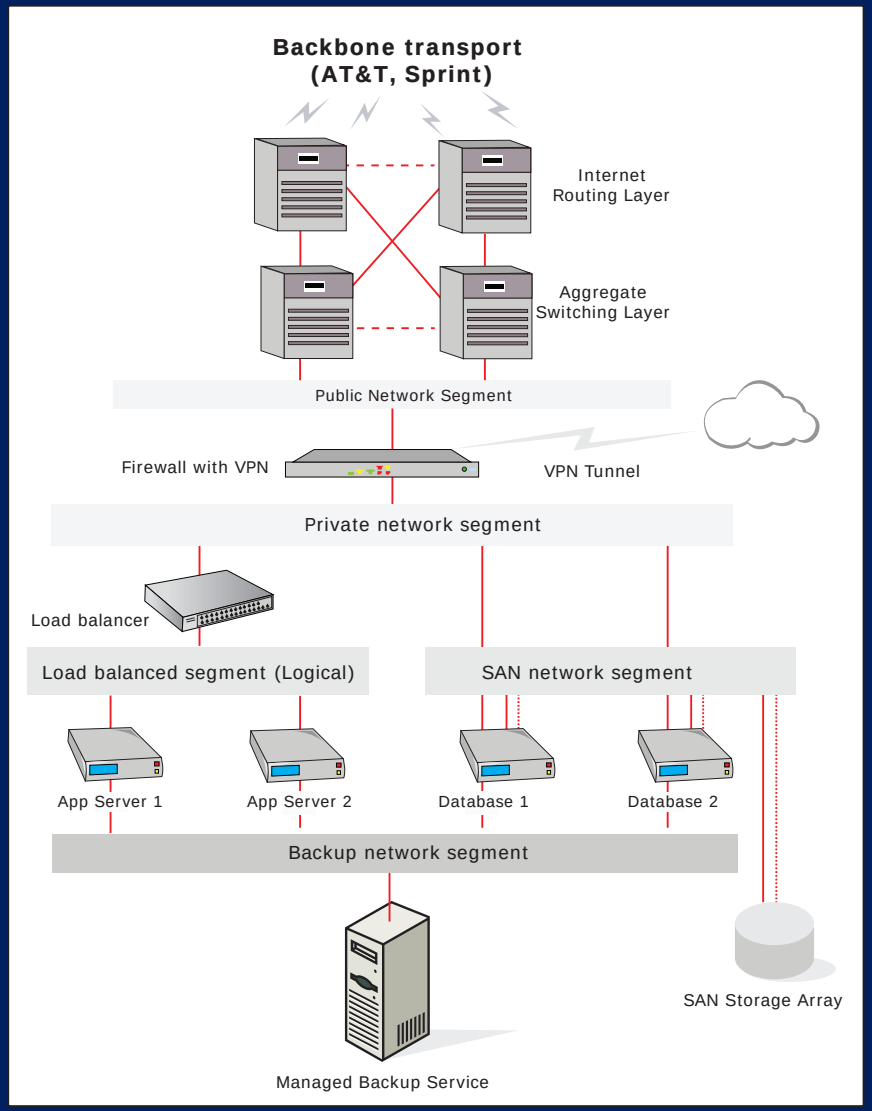


Image by MIT OpenCourseWare.

Router, firewall, load balancer

- Router routes IP (TCP/IP and UDP/IP) packets
  - Routing table: send packet to local machine or to another router
- Firewall examines all packets
  - Checks IP address, port
  - Extracts contents from all Internet protocols if suspicious
- Load balancer sends HTTP packets to Web servers
  - Balances based on HTTP header: client IP, cookie, …
  - Many inexpensive servers function as single server
  - Add new servers, sites without interruption to service

Web server, app server, database server

- Web (HTTP) server:
  - Handles HTTP requests and responses
  - Reads and writes XML and XHTML documents
- Application server:
  - Think of it as a pre-written main program with many pre-written components for common tasks
    - Can generate Web pages/XML, read/write from db easily
  - Designed from UML use cases, sequence diagrams: e.g.,
    - Order entry, receiving, shipping, payment, engineering, …
- Database server:
  - Persistent storage of data
  - Sharing of data across applications and organizations
  - Repository of business rules, within database and to guide XML document exchange

System architecture and configuration

- Organization of all system components (hardware, software, network) is the system architecture
- Done early in system development or configuration project
  - Assess users, applications, system software, networks, hardware
  - Configure Web/app/db servers, networks, backup, etc.
  - Configurations are complex and changing
  - Information is usually wrong on which estimates are based
  - Step is needed because you need a budget for a project
- Remember the estimate convergence curve
  - There is much uncertainty in the final system hardware and software. We are only in spiral 1 still.
  - If you make a baseline (point) estimate, you have a lot of error and about a 50% chance of being too low
  - If you've done everything else right but your system is slow because of inadequate hardware or telecom, it will be very frustrating
  - Your customer can't tell why your system is slow. It may be bad database, bad/inefficient software implementation, etc.
  - This is one of the few problems that you can throw money at to solve!

System architecture and configuration, p.2

- Successful configurations are usually overbuilt
  - Most components will be oversized, but you have a better chance of having enough capacity at the (unanticipated) bottleneck
  - Spend your entire budget, always, or use cloud computing
- Each server (often virtualized) does just one thing:
  - HTTP server (Web server)
  - Business logic (app server)
  - SQL (database server)
  - We can understand and characterize its task this way. If a box handles many functions, sizing and managing it is hard

1.264J / ESD.264J Database, Internet, and Systems Integration Technologies
Fall 2013