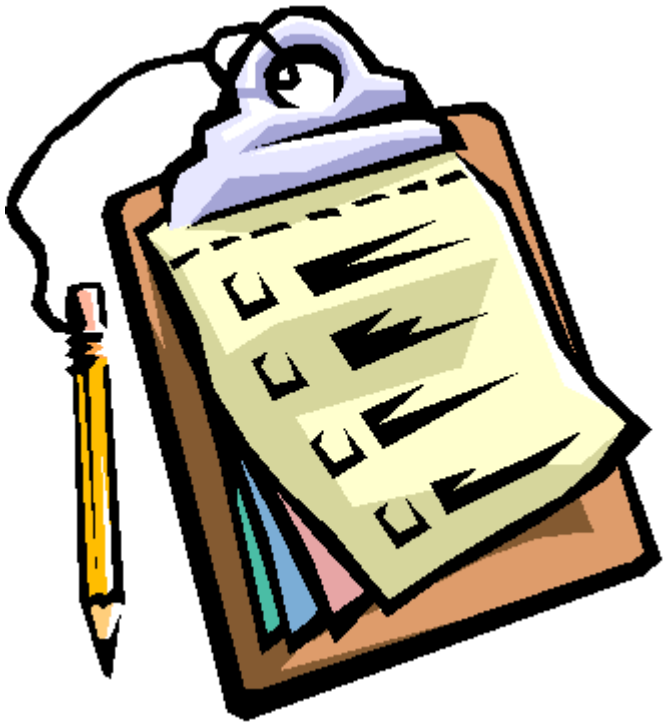
Internet Engineering Course

Web Application Architecture

Sadegh Aliakbary

Agenda

- WebApp Architecture
- WebApp Layers
- MVC Architectural Design Pattern
- Service Oriented Architecture



Software Architecture

- The structure of the system, which comprise:
  - software components,
  - the externally visible properties of those components,
  - and the relationships among them

- A high-level software design

Why is Architecture Important?

- An enabler for **communication**
  - between all stakeholders
- The architecture **highlights early design decisions**
  - will have a profound impact
    on all following software engineering work
- Architecture "*constitutes a **relatively small**, intellectually graspable mode of **how the system is structured** and how its components work together*"

Architectural Decisions

- The system architect considers a variety of **alternatives**,
- and ultimately decides on the specific architectural features that best meet the requirements
- E.g., Styles, Patterns, …
  - Usually considered in design (refinements): Technologies, …
- E.g.,
  - Client-Server Architecture, Java-based design, .NET-based design
- Major decisions are traceable in architecture

Web Application Layers

Software Layer

- A layer is a group of reusable components that are reusable in similar circumstances

- Layer vs Component (Module)
  - A layer may encompass some components
  - Components of a layer: restricted to the tasks of that layer

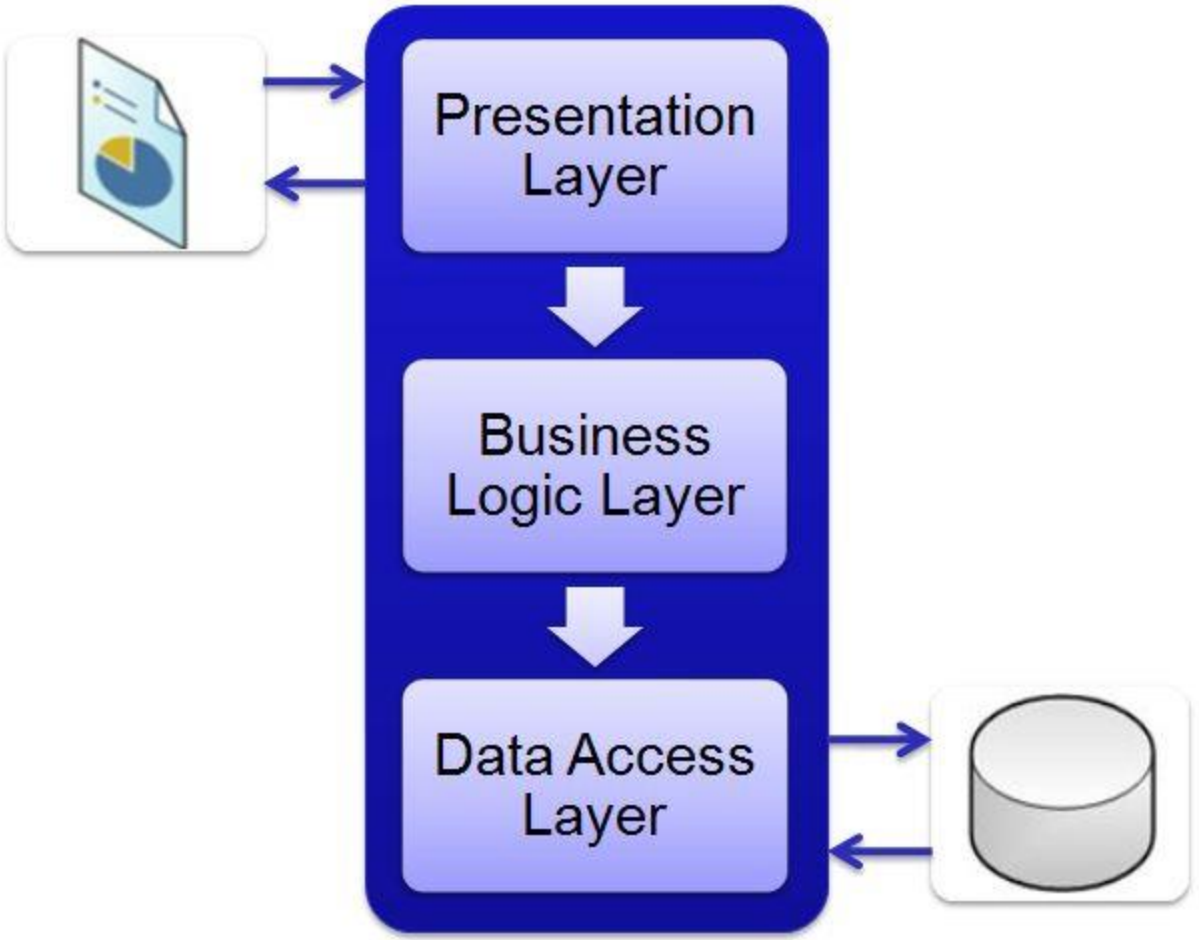
- Layer vs tier (Multi-layer vs Multi-tier)
  - Layer is logical, Tier is physical

Common Layers

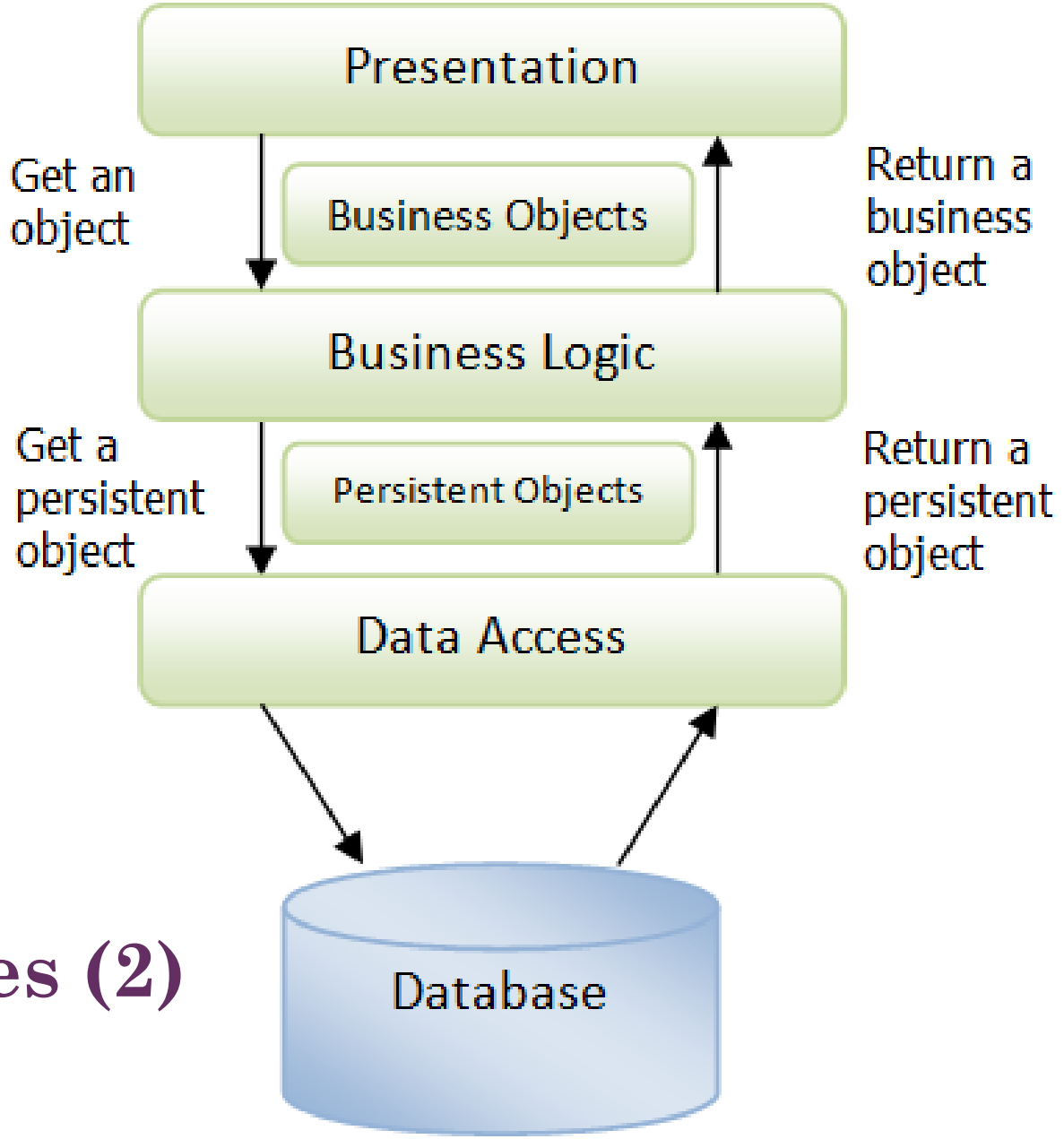
- Presentation layer
  - UI, view
- Service layer
  - service
- Business layer
  - business logic, domain layer
- Data access layer
  - persistence layer

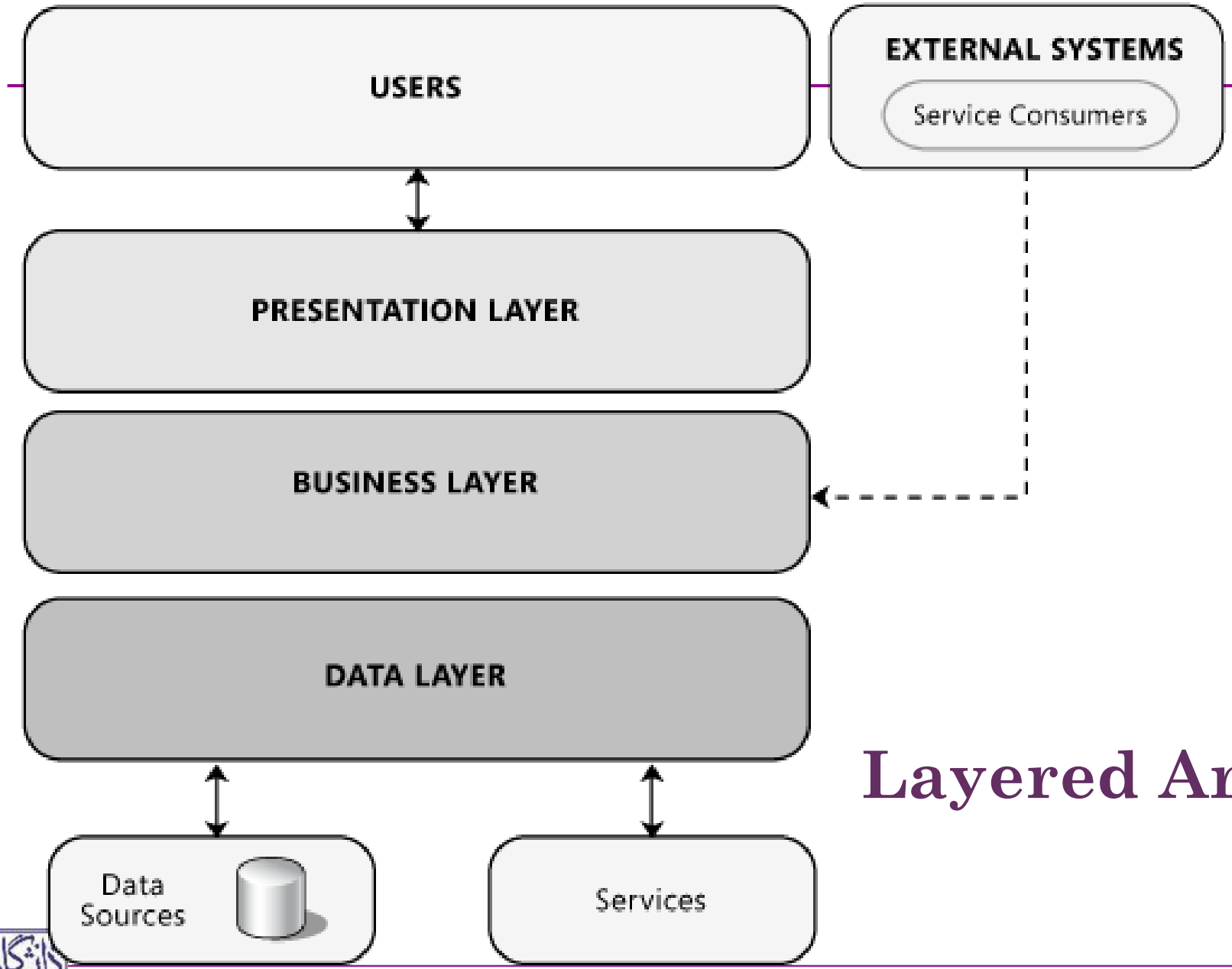
- Three layer architecture:
  - Presentation
  - Business Logic (service + business)
  - Data Access

Sample Layered Architectures (1)



Layered Architectures (2)

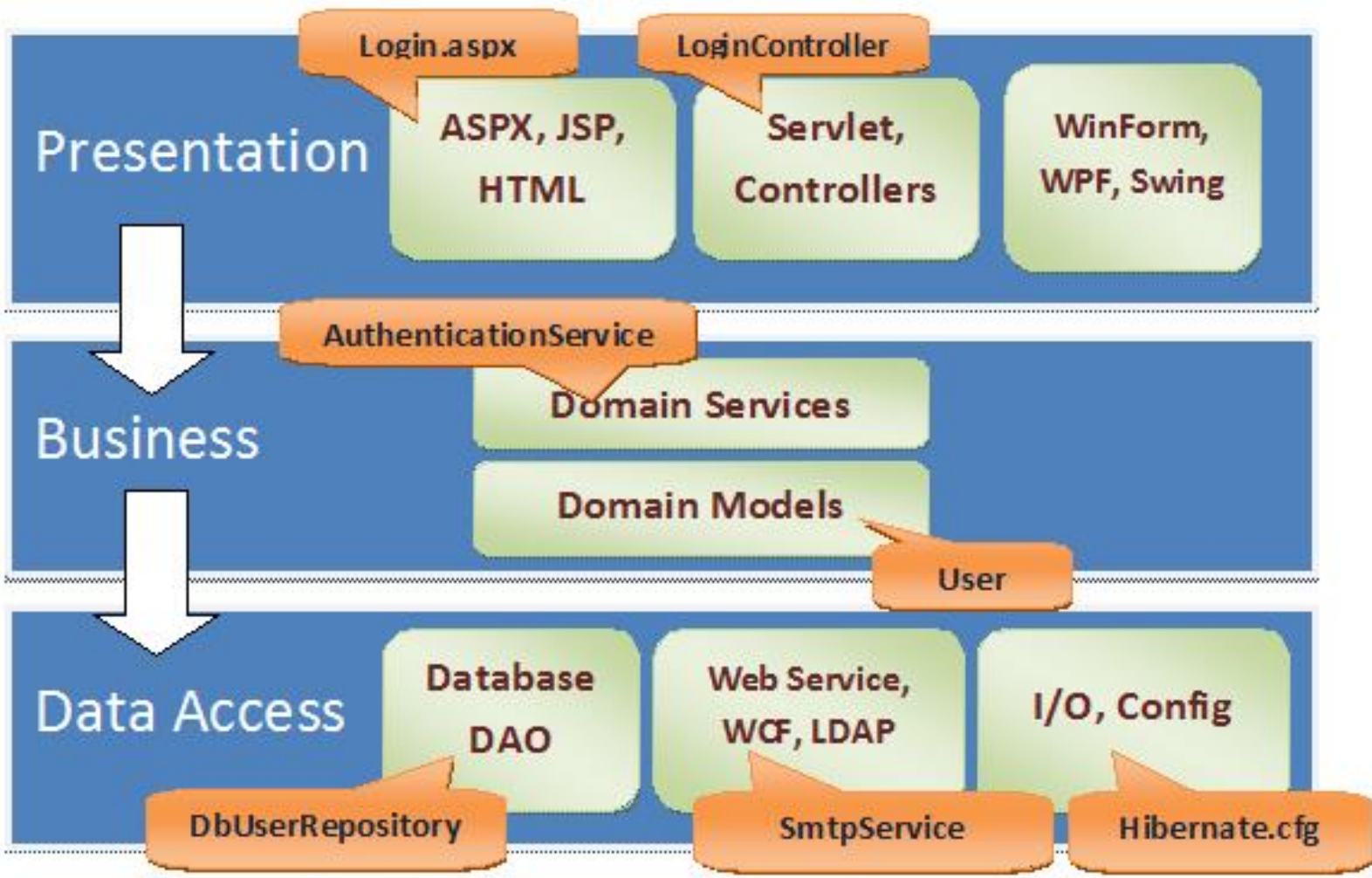




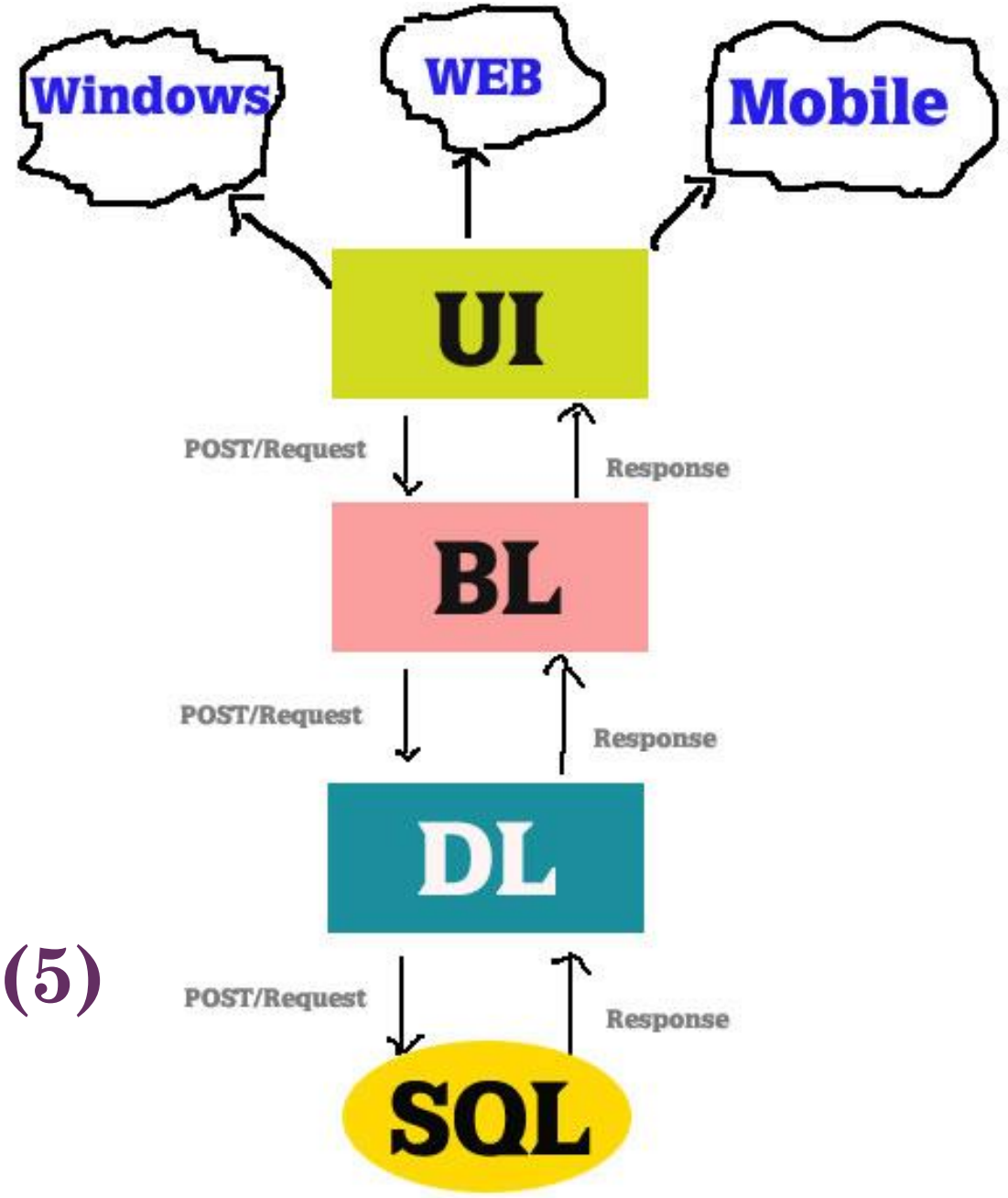
Layered Architectures (3)

Layered Architectures (4)

- Find the layer:
  - HTML
  - Javascript
  - Dataset
  - calculate()
  - db.commit()



Layered Architectures (5)



No Layering Example

```
<sql:query var="books" dataSource="${datasource}">
  SELECT id, title, price FROM book
</sql:query>

<html>
 <body>
    <table border="1">
      <tr>
        <td>id</td><td>title</td><td>price</td>
      </tr>
      <c:forEach items="${books.rows}" var="row">
      <tr>
        <td><c:out value="${row.title}" /></td>
        <td><c:out value="${row.price}" /></td>
      </tr>
      </c:forEach>
    </table>
  </body>
</html>
```

Layering, Pros and Cons

- Pros
  - Manageable (extensible) software
  - Developers often focus on particular skills
- Cons
  - Cost of communication
  - Restricted layer API

MVC

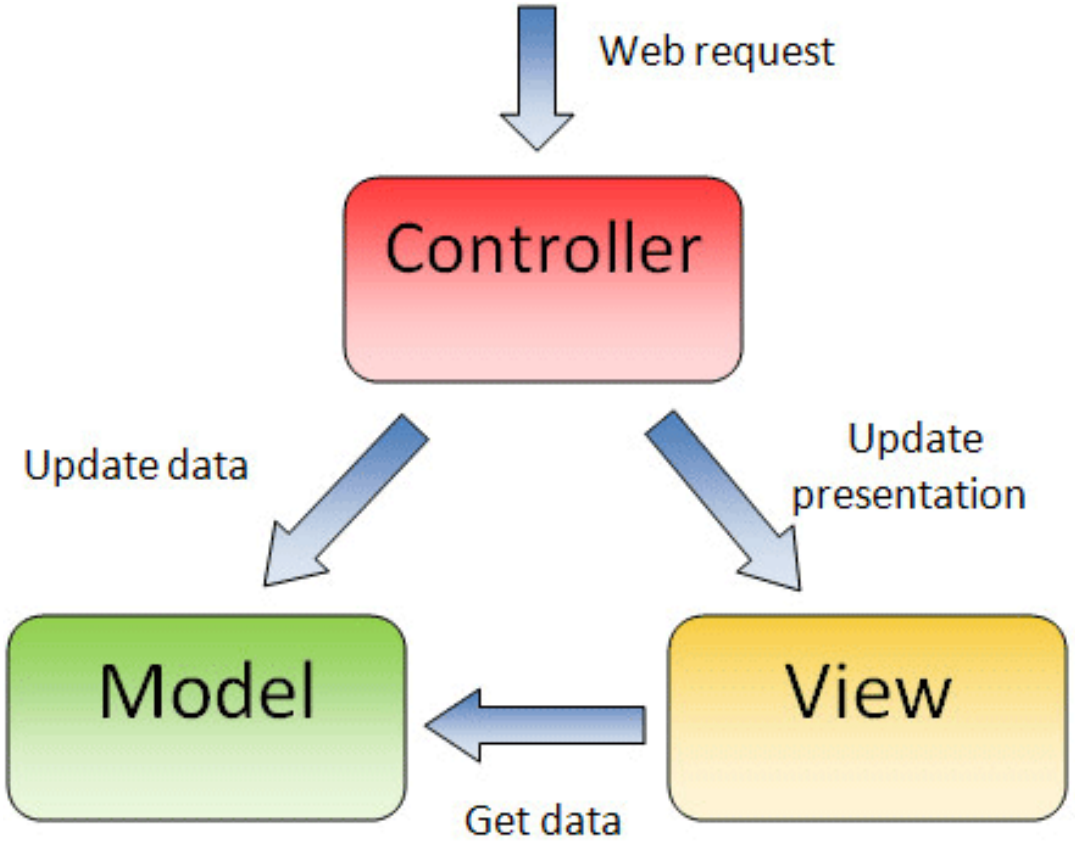
MVC Architectural Pattern

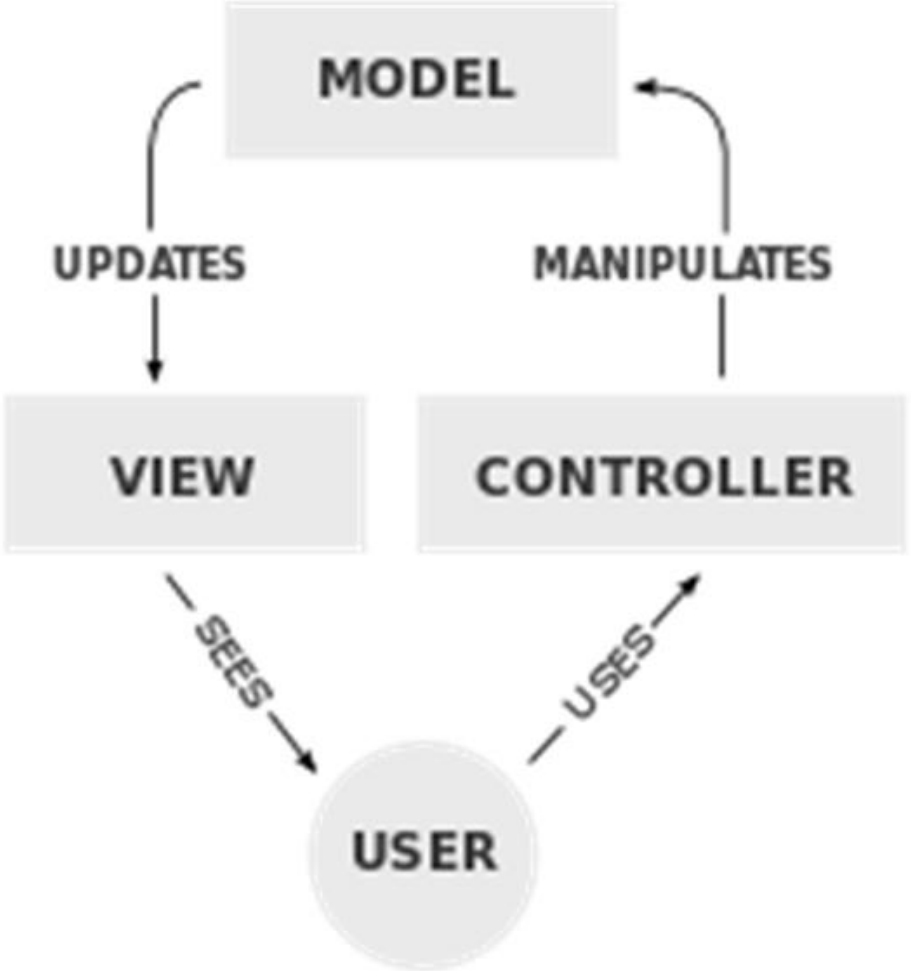
- A software **architectural pattern** for implementing user interfaces
- It divides a given software application into three interconnected parts:
  - **Model, View, Controller**
  - separates internal representations of information from the ways that information is presented to or accepted from the user

MVC Pattern

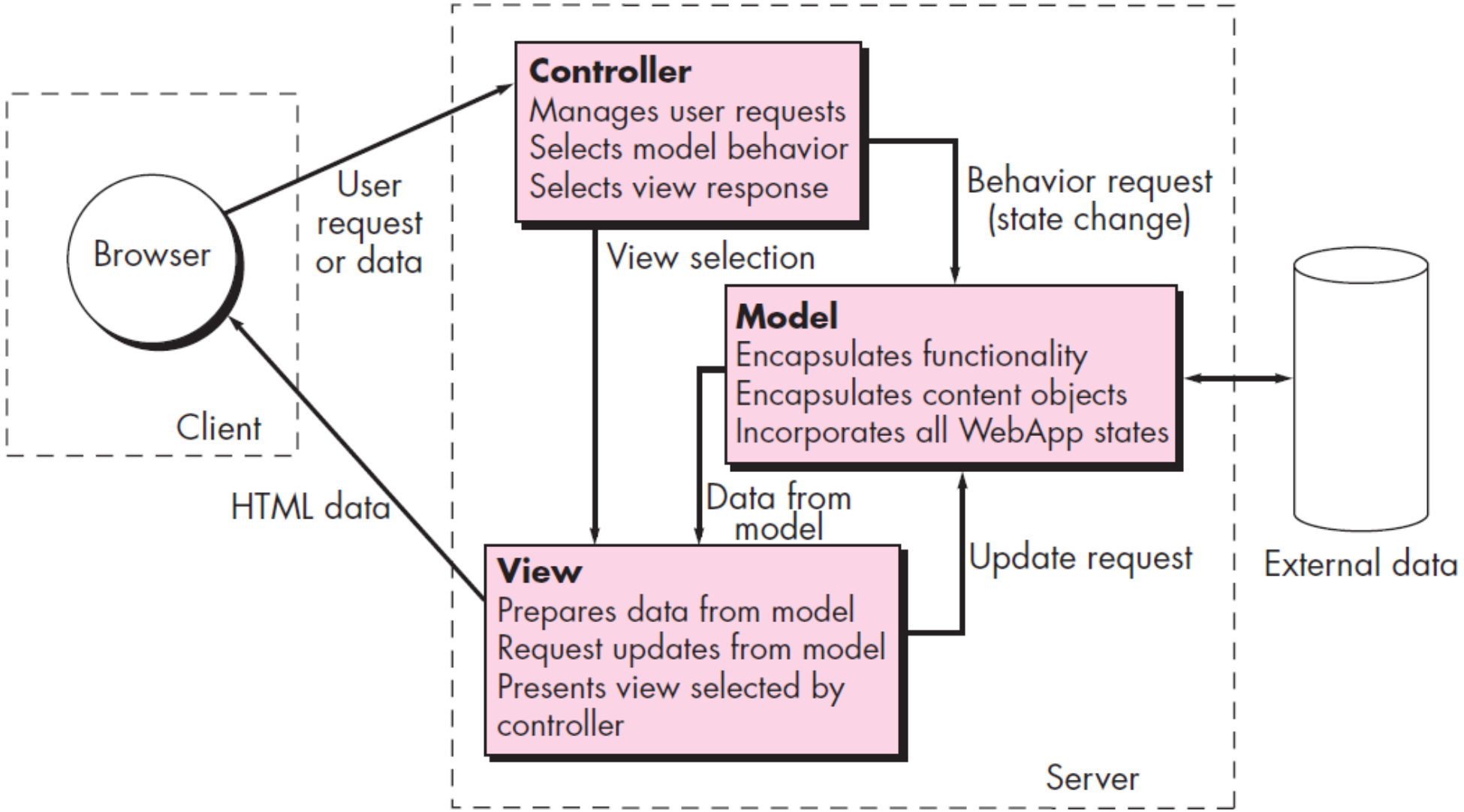
- **The model** contains all application specific content and processing logic, including
  - all content objects
  - access to external data/information sources,
  - all processing functionality that are application specific
- **The view** contains all interface specific functions and enables
  - the presentation of content and processing logic
  - all processing functionality required by the end-user.
- **The controller** manages access to the model and the view and coordinates the flow of data between them.

MVC

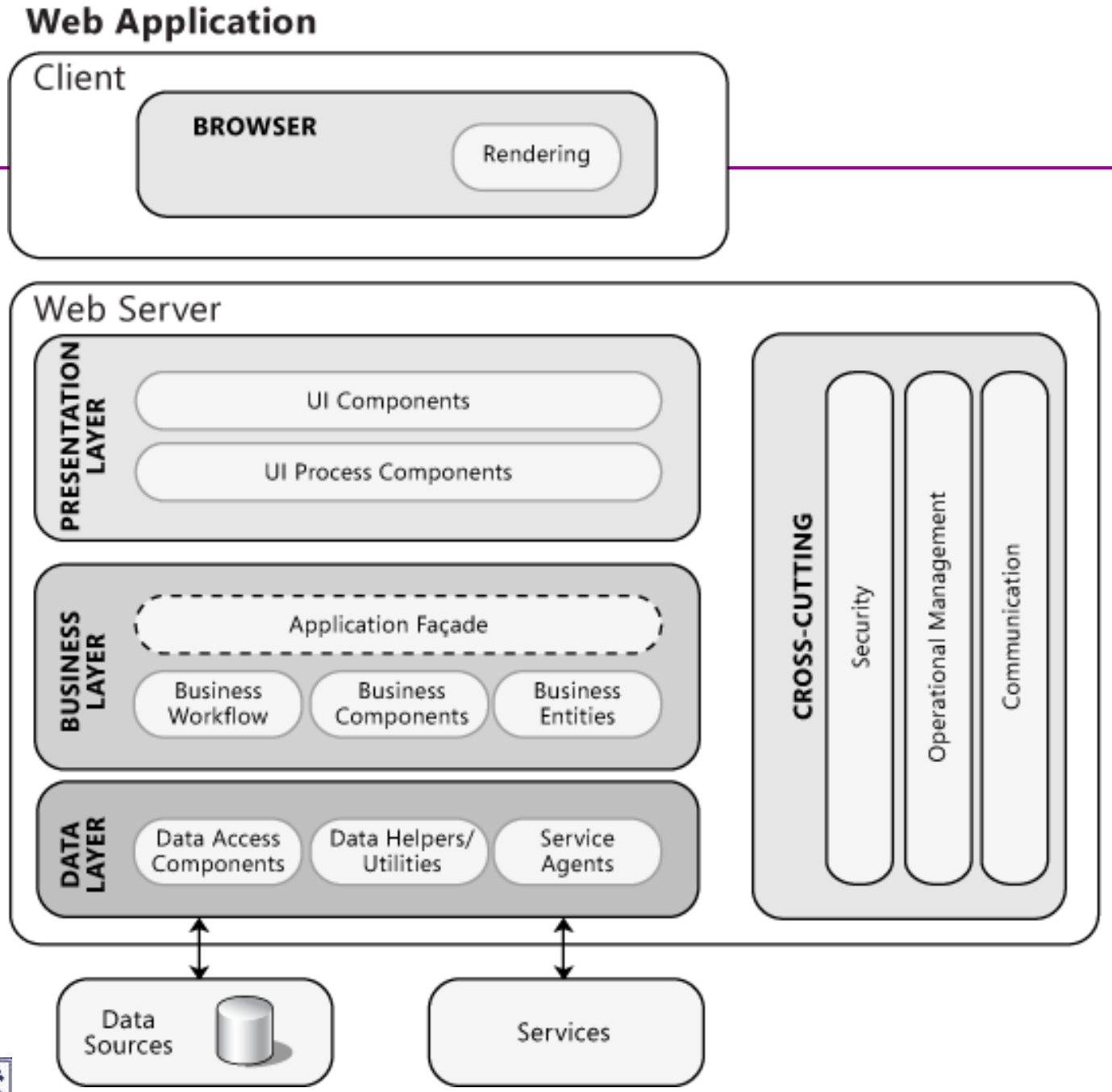

Web request
Controller
Update data
Update presentation
Model
View
Get data


MODEL
UPDATES
MANIPULATES
VIEW
CONTROLLER
SEES
USES
USER

MVC Architecture



A Typical WebApp Architecture

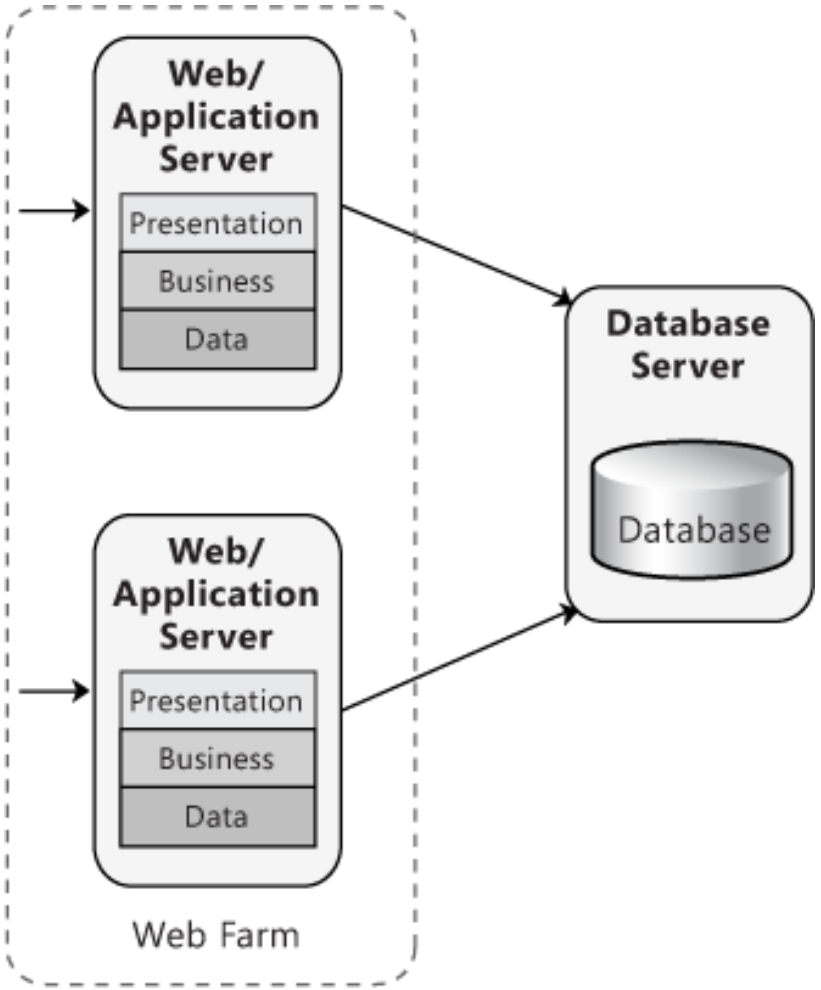


Notes on WebApp Design

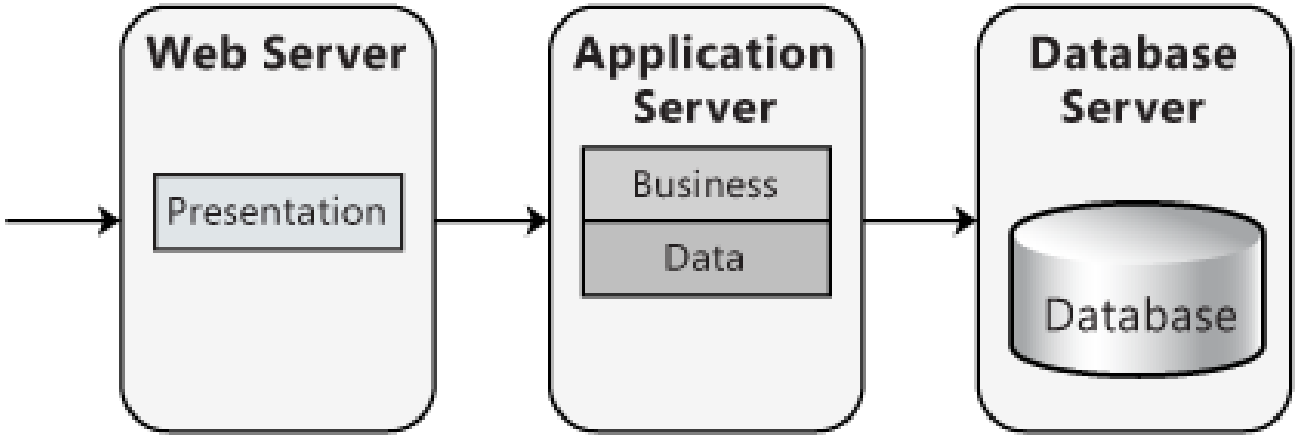
Specific Design Issues

- Application Request Processing
  - Validation
- Authentication & Authorization
- Caching
- Logging
- Navigation
- Page Layout & Rendering
- Session Management

Physical View Examples



Load Balancing



Distributed Deployment

Service Oriented Architecture

What is a Service Oriented Architecture (SOA)?

- A method of design, deployment, and management of both applications and the software infrastructure where:
  - Software is **organized into business services** that are **network accessible** and **executable**
  - Service interfaces are based on public standards for **interoperability**

What is a “Service”?

- A Service is a reusable component.
- Changes business data from one state to another.
- A Service is the only way how data is accessed.

What is Web Service?

- A service provided on the web protocols
- Similar to a web page
  - A web page is consumed by a user (browser)
  - A web service is consumed by an application (software)
- Web services are based on standards
  - SOAP (Simple Object Access Protocol)
  - **REST** (Representational State Transfer)

SOAP vs REST

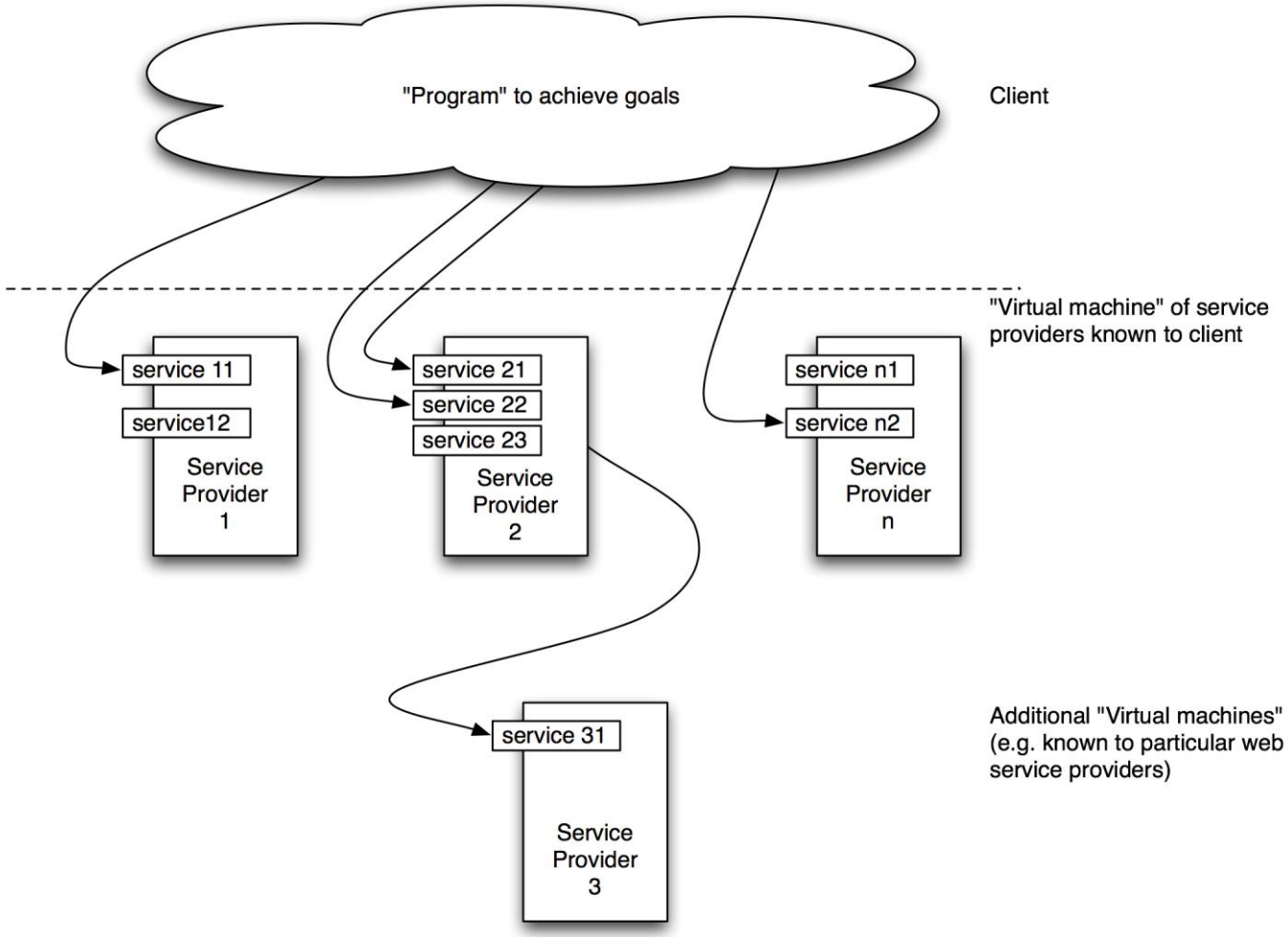
- SOAP is based on XML
  - WSDL is an example that describes the service
  - SOAP messages are based on XML

- REST web services supports JSON

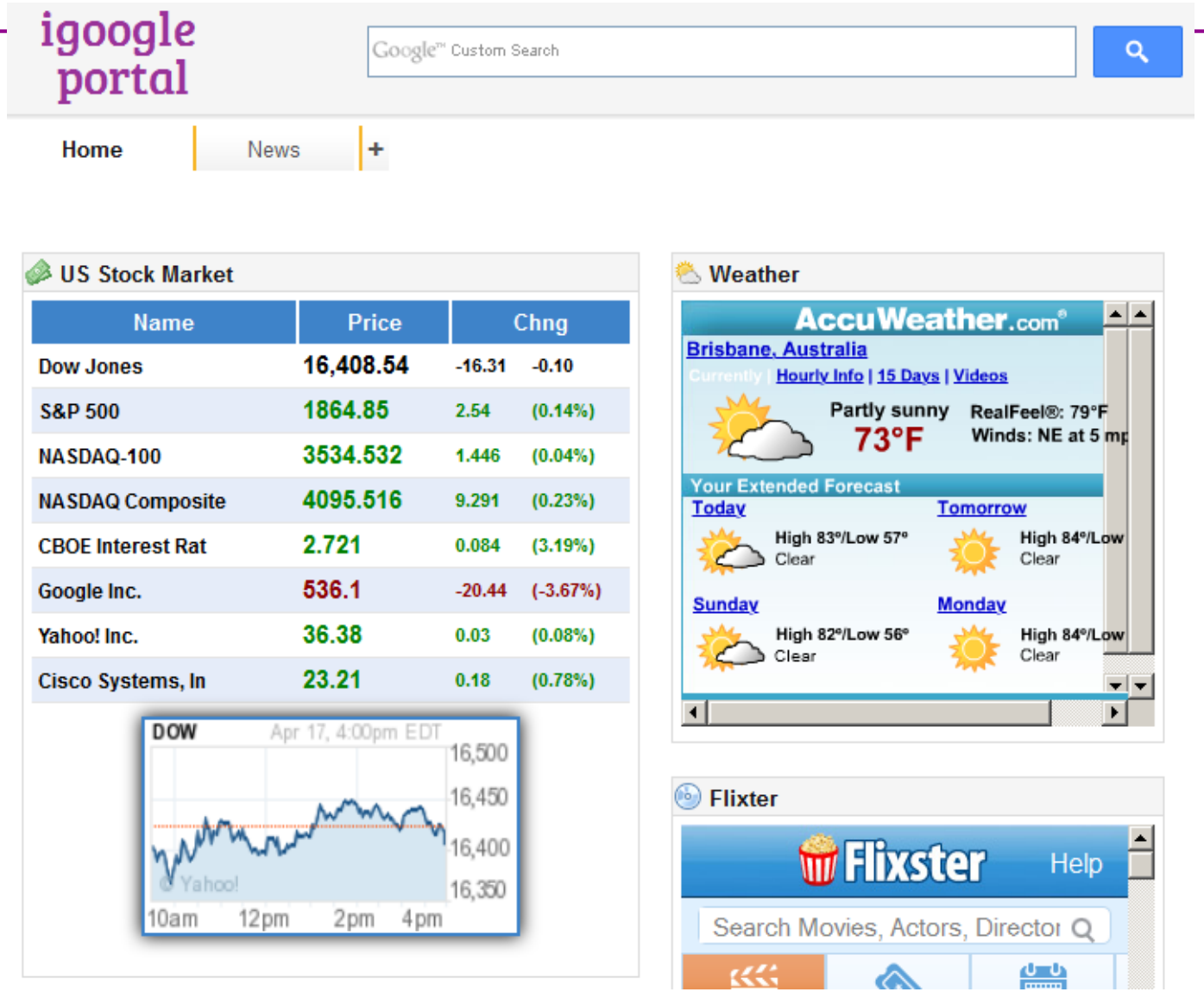
Sample Services in a Web Application

- Authenticate(username, password)
- Authorize(user, accessed_form)
- Persist(user)
- findProduct(product-info)
- startLoanWorkflow(user, amount-of-loan)

Web services on the Internet



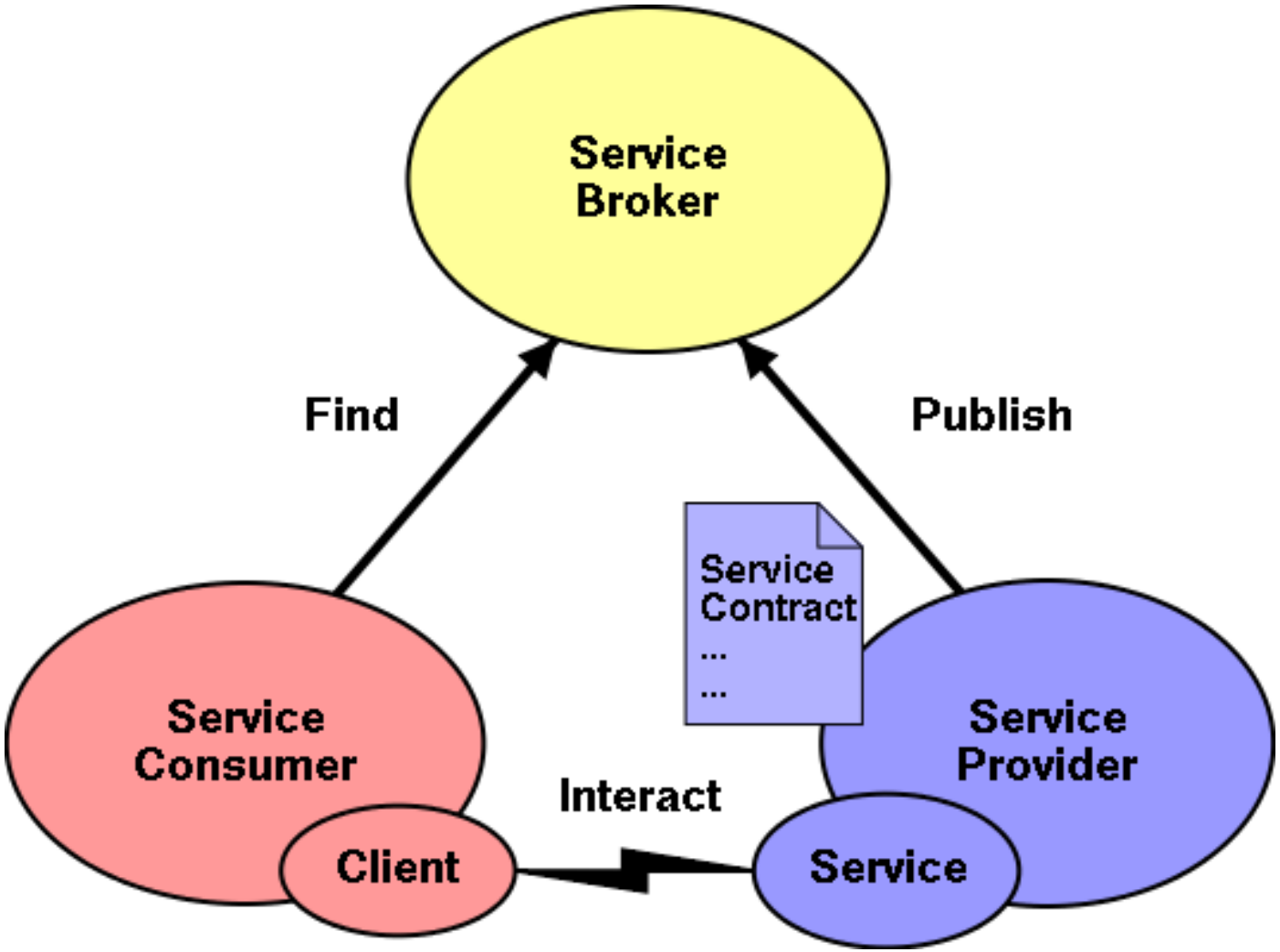
iGoogle


igoogle portal
Google™ Custom Search
Home
News
+
US Stock Market
Name
Price
Chng
Dow Jones
16,408.54
-16.31
-0.10
S&P 500
1864.85
2.54
(0.14%)
NASDAQ-100
3534.532
1.446
(0.04%)
NASDAQ Composite
4095.516
9.291
(0.23%)
CBOE Interest Rat
2.721
0.084
(3.19%)
Google Inc.
536.1
-20.44
(-3.67%)
Yahoo! Inc.
36.38
0.03
(0.08%)
Cisco Systems, In
23.21
0.18
(0.78%)
DOW
Apr 17, 4:00pm EDT
16,500
16,450
16,400
16,350
Yahoo!
10am
12pm
2pm
4pm
Weather
AccuWeather.com
Brisbane, Australia
Currently | Hourly Info | 15 Days | Videos
Partly sunny
73°F
RealFeel®: 79°F
Winds: NE at 5 mp
Your Extended Forecast
Today
High 83°/Low 57°
Clear
Tomorrow
High 84°/Low
Clear
Sunday
High 82°/Low 56°
Clear
Monday
High 84°/Low
Clear
Flixter
Flixster
Help
Search Movies, Actors, Director

Interoperability Alternatives

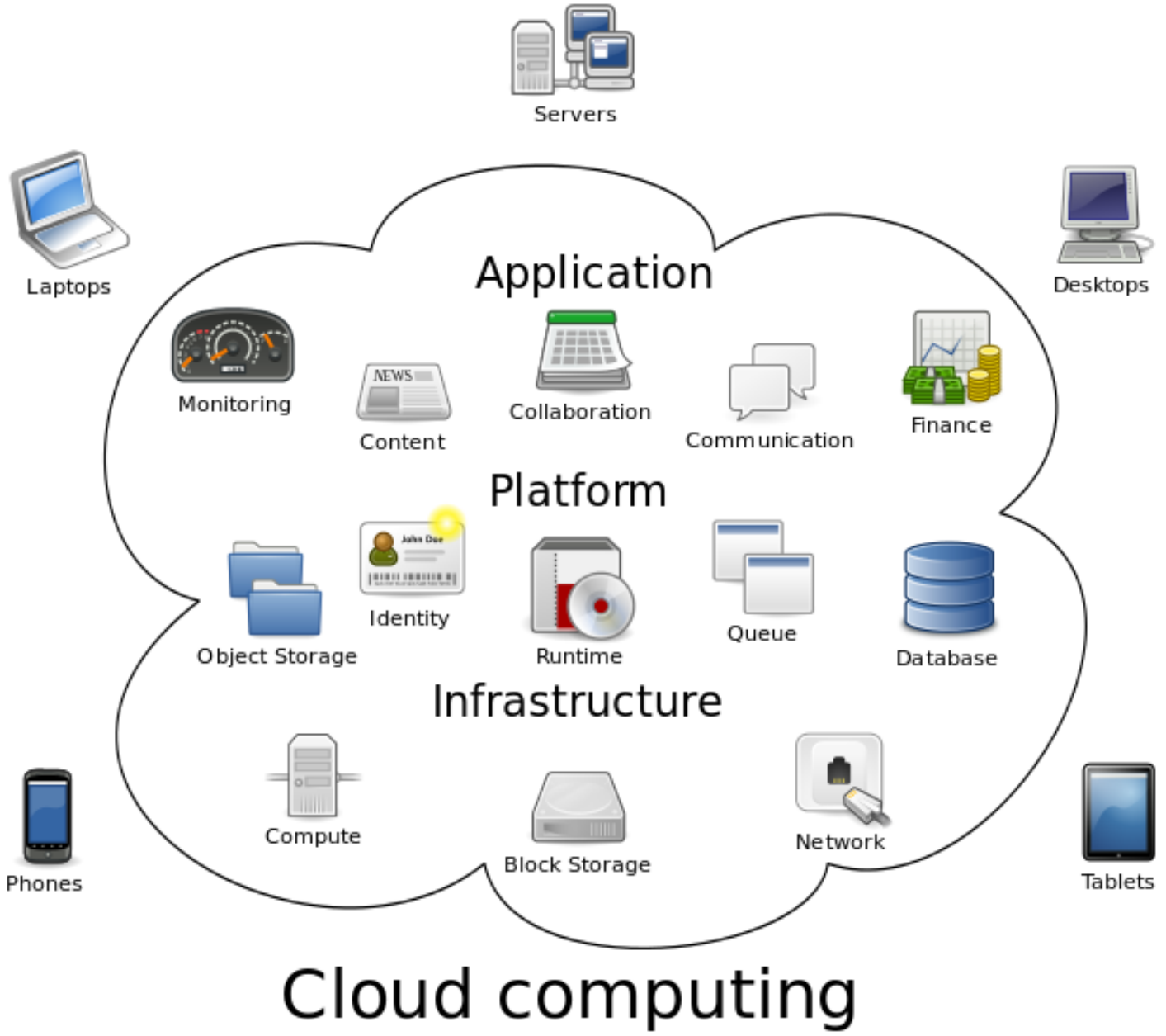
- Direct method invocation vs remote method invocation
  - Distributed processing
  - Scalability
- RMI (RPC) vs Services
  - Platform independence

Service Broker



Cloud Computing

Cloud


Servers
Laptops
Desktops
Application
Monitoring
Content
Collaboration
Communication
Finance
Platform
Object Storage
Identity
Runtime
Queue
Database
Infrastructure
Compute
Block Storage
Network
Phones
Tablets
Cloud computing

Cloud Computing

- On-demand computing
- A kind of internet-based computing
- Provides shared processing resources on demand
  - shared pool of configurable computing resources
  - e.g., networks, servers, storage, applications and services
  - Rapidly provisioned/released with minimal management effort
- Store and process their data in third-party data centers
  - Resource sharing to achieve coherence and economy of scale

Cloud Computing Benefits

- Economy (save money!)
  - Both for cloud customers and cloud providers
  - High computing power, with cheap cost of services
  - Scalability
  - Pay as you go (Pay per use)
- Allows companies to avoid upfront infrastructure costs
  - They **focus on projects** that differentiate their businesses
  - instead of on infrastructure

Cloud Enablers

- High-capacity networks
- Low-cost computers and storage devices
- Hardware virtualization
  - Cloud vs Hosting?
  - Cloud vs Virtualization?
- Service-oriented architecture

Elasticity

- Companies can scale up as computing needs increase and then scale down again as demands decrease
- **Dynamic** ("on-demand") provisioning of resources
  - on a fine-grained, **self-service** basis
  - in near real-time

Characteristics of Cloud Computing

- Agility
- Cost
- Device and location independence
- Maintenance
- Performance
- Scalability and elasticity
- Security

Cloud Summary

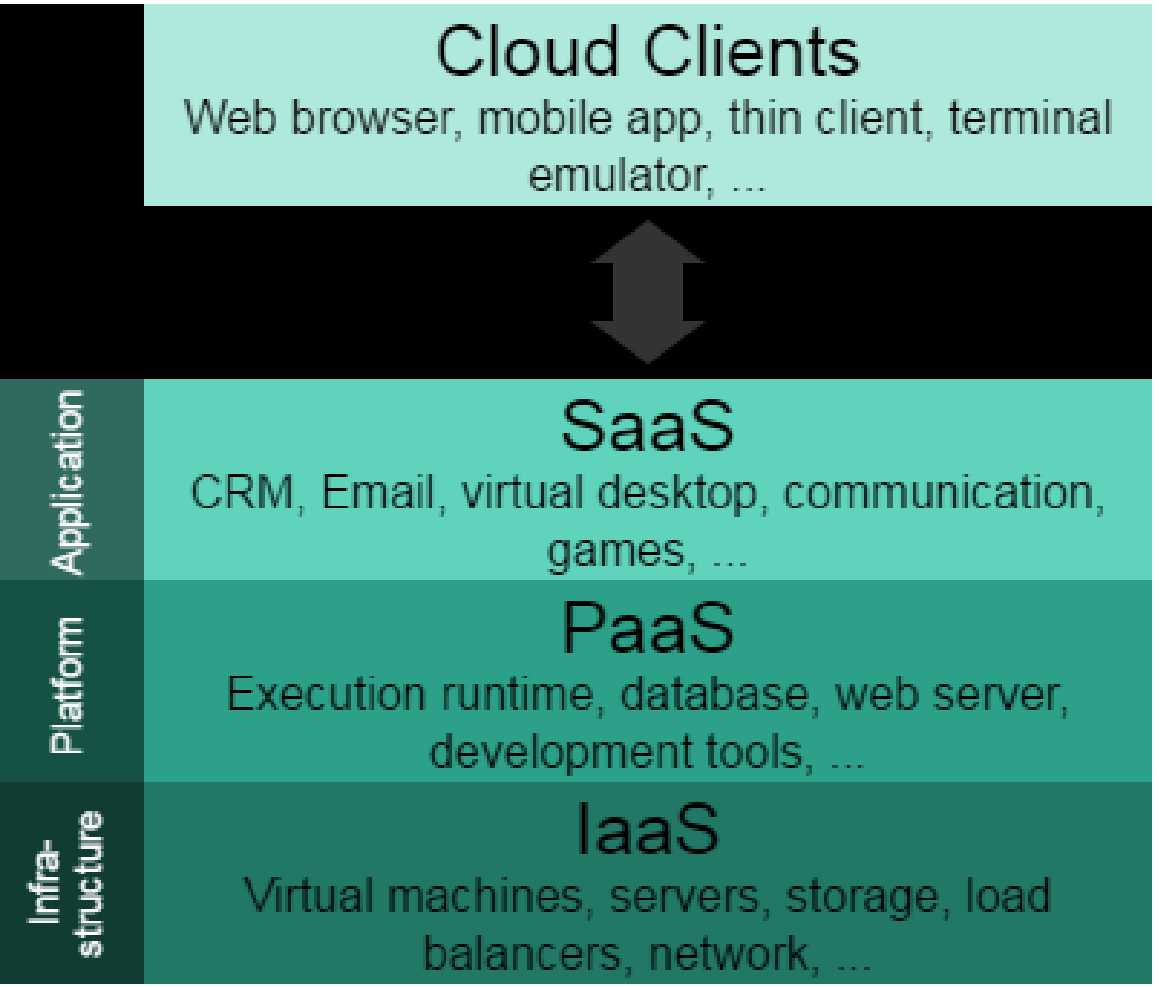
- Self-service provisioning
- Elasticity
- Pay per use

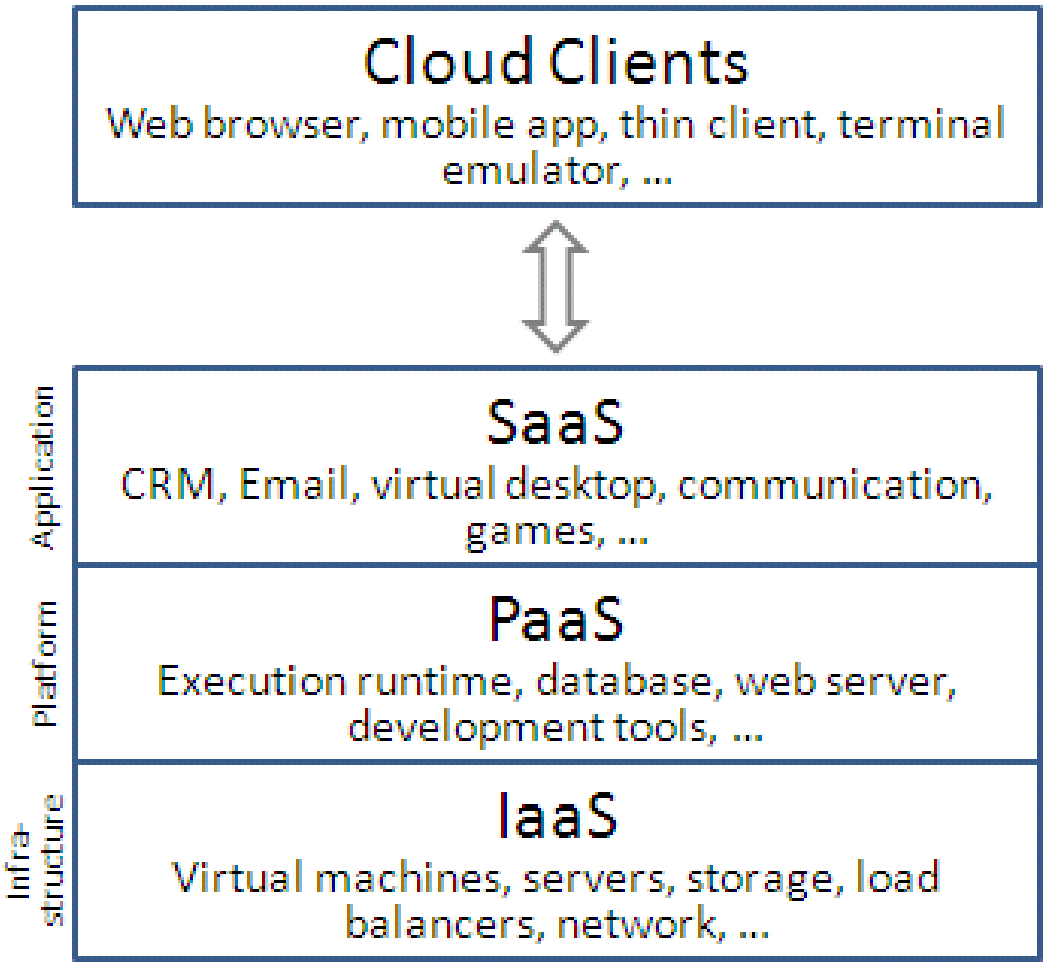
Service Models

- Infrastructure as a service (IaaS)
- Platform as a service (PaaS)
- Software as a service (SaaS)

- In this context, a service is not [necessarily] a web-service

Service Models





Infrastructure as a service (IaaS)

- Offers:
  - Physical or (more often) virtual machines
  - Storage
  - And other resources.
- Online services
  - that abstract the user from the details of infrastructure
  - like physical computing resources, location, data partitioning, scaling, security, backup etc.
- Examples?!

Platform as a service (PaaS)

- Offer a development environment
  - to **application developers**
- Deliver a computing platform, including:
  - Operating system
  - Database
  - Web server
- The consumers, are the developers
- Examples: Amazon, Google, and Microsoft clouds

Software as a Service

- End users gain access to application software
- On-demand software
- Usually priced on a **pay-per-use** basis or using a subscription fee
- Gives a business the potential to reduce IT operational costs
  - by outsourcing hardware and software maintenance and support to the cloud provider.
- Example?

See Also:

- Open Source software for creating clouds
  - OpenStack
- Cloud successful examples
  - Salesforce
  - Amazon
  - Dropbox
  - …

The End