

ATI – Arsitektur Aplikasi Web

anton@ti.ukdw.ac.id



Internet & Web Basics



Internet

- What is Internet?
 - **A world-wide network of computer networks**
- Internet is **huge** client server system

Internet vs. WWW

Internet is the infrastructure that makes the WWW work.

- **Packet Switching**
- **TCP/IP Protocol**
- **Physical Infrastructure**
 - Fiber-optics lines, wires
 - Satellites, Cable Modems
 - Routers, Hubs, Network Cards, WiFi systems, etc.

WWW is just one of many “virtual networks” built on the Internet.

- **Websites:** http, https, etc.
- Email: pop, imap, etc.
- Other systems: ftp, instant messaging, etc.



Web

- **Tim Berners-Lee** at CERN proposed the Web in 1989
 - **Purpose:** to allow scientists to have access to many databases of scientific work through their own computers
- From document (**linear media**) to hypertext (**hypermedia**)
 - Ex of Linear media is a book, pages, document, cassette
 - We'll call them **documents**
 - Hypermedia – media that provide point to the need
 - Examples: CD, DVD, Webpages
 - Web provides **Hyperlink**

Protokol Web: HTTP

- **Hypertext** Transport Protocol (RFC 1945)
 - Oleh Tim Berners-Lee, 1990
- Protocol that used to communicate between web browsers and web servers
- Using TCP port **80** (default)
- This protocol supports *hypermedia* files
- HTTP 1.0 becomes 1.1 by IETF (RFC 2616)



HTTP Session

- When client and server communicate, their **connection** has to be **maintenance...**
- HTTP provides **session**
- A basic **HTTP session** has four phases:
 - Client **opens the connection**
 - Client makes the **request**
 - Server sends a **response**
 - Server **closes the connection**

HTTP Stateless

- But HTTP is “**stateless**”
 - HTTP is a **stateless protocol**, which means that once a server has delivered the requested data to a client, the connection is broken, and the server retains no memory of what has just taken place
- **No information that retain**
- **Solution? Cookies & Session**

HTTP (2)

- HTTP is **request – response**:
 - HTTP client (**user agent**) requests to HTTP server and Server will response
- The basic different of HTTP/1.1 and HTTP/1.0 is their **persistent connection**
- HTTP/1.0 is **not** using persistent connection
 - **Header = Connection: close**
- HTTP/1.1 is using **persistent** connection
 - **Header = Connection: Keep-Alive**
 - But client can set header manually (header = Connection: close).

```
Mark ~
bitmask@x220 ~
$ telnet www.odetocode.com 80
Trying 96.31.33.25...
Connected to www.odetocode.com.
Escape character is '^]'.
GET / HTTP/1.1
host:www.odetocode.com

HTTP/1.1 301 Moved Permanently
Location: http://odetocode.com/
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Date: Fri, 13 Jan 2012 23:21:24 GMT
Connection: close
Content-Length: 0

Connection closed by foreign host.

bitmask@x220 ~
$
```

Connection

```
CONNECT www.bad.carbonwind.net:443 HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; SU1; .NET CLR 1.1.4322)
Host: www.bad.carbonwind.net
Content-Length: 0
Proxy-Connection: Keep-Alive
Pragma: no-cache

HTTP/1.1 200 Connection established
Via: 1.1 LAB4ISA
Connection: Keep-Alive
Proxy-Connection: Keep-Alive
```

HTTP Message

HTTP-message = Request | Response ; HTTP/1.1 messages

generic-message = start-line
*(message-header CRLF)
CRLF [message-body]

start-line = Request-Line | Status-Line

message-header = field-name ":" [field-value] *'host:'* mandatory

Request-Line = Method SP Request-URI SP HTTP-Version CRLF

Method = "OPTIONS" ; Section 9.2
| "GET" ; Section 9.3
| "HEAD" ; Section 9.4
| "POST" ; Section 9.5
| "PUT" ; Section 9.6
| "DELETE" ; Section 9.7
| "TRACE" ; Section 9.8
| "CONNECT" ; Section 9.9
| extension-method

HTTP Request

- A request consists of:
 - Initial line
 - Headers
 - Blank line
 - Message body

Request Example

GET /courses/ati/index.html HTTP/1.1

From: anton@ukdw.ac.id

User-Agent: Firefox

[blank line here]

Method

Path

Version

Headers

Initial line

HTTP Request Methods

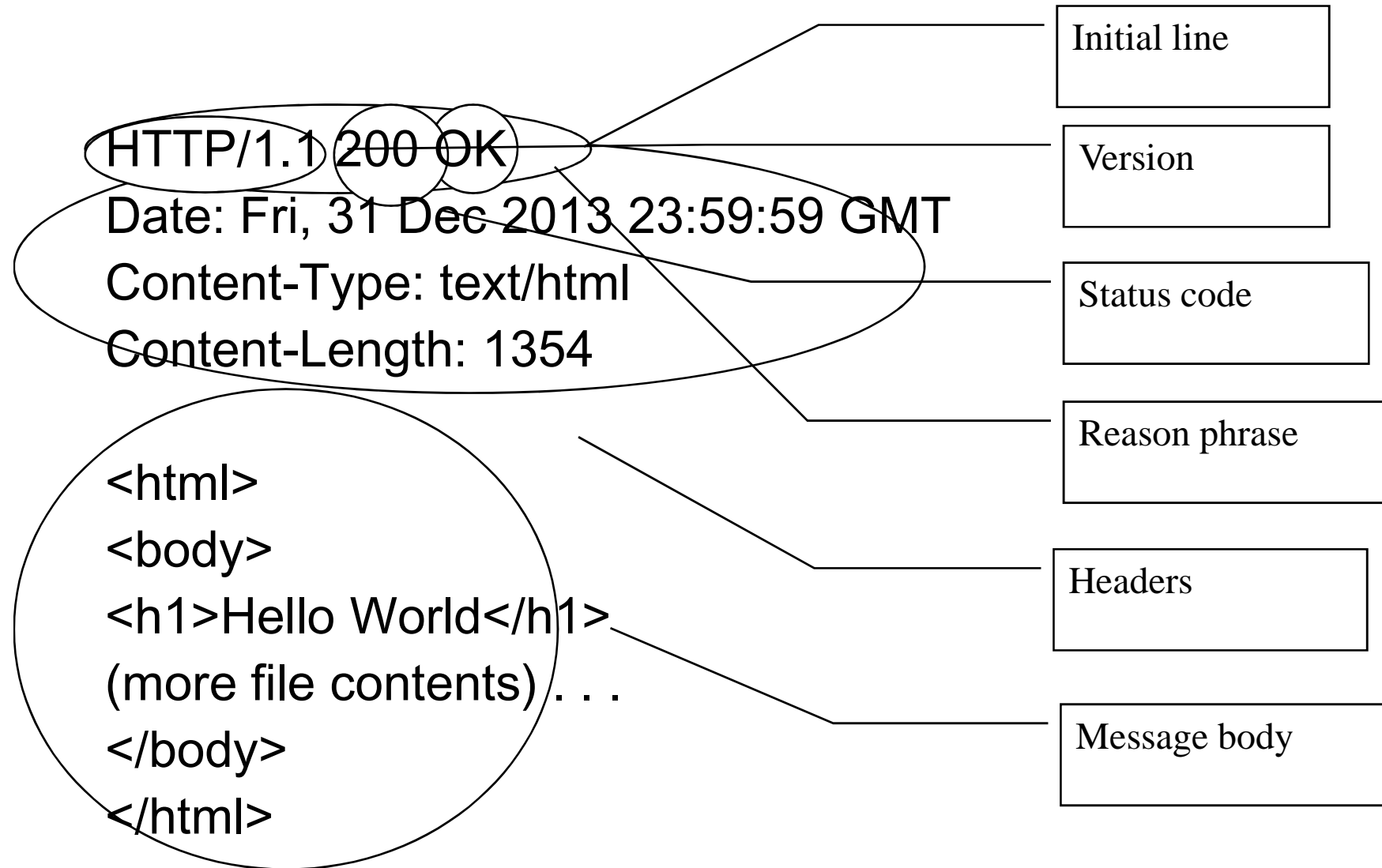
- **GET** - Fetch a document
 - Ex: URL variabel
- **POST** - Execute the document, using the data in body document
 - Ex: form submit
- **HEAD** - Fetch just the header of the document
 - Ex: date modified
- **PUT** - Store a new document on the server
 - Ex: upload using WebDAV
- **DELETE** - Remove a document from the server

HTTP Response status

Response Code	HTTP Operation	Response Body Contents	Description
200	GET, PUT, DELETE	Resource	No error, operation successful.
201 Created	POST	Resource that was created	Successful creation of a resource.
202 Accepted	POST, PUT, DELETE	N/A	The request was received.
204 No Content	GET, PUT, DELETE	N/A	The request was processed successfully, but no response body is needed.
301 Moved Permanently	GET	XHTML with link	Resource has moved.
303 See Other	GET	XHTML with link	Redirection.
304 Not Modified	conditional GET	N/A	Resource has not been modified.
400 Bad Request	GET, POST, PUT, DELETE	Error Message	Malformed syntax or a bad query.
401 Unauthorized	GET, POST, PUT, DELETE	Error Message	Action requires user authentication.
403 Forbidden	GET, POST, PUT, DELETE	Error Message	Authentication failure or invalid Application ID.
404 Not Found	GET, POST, PUT, DELETE	Error Message	Resource not found.
405 Not Allowed	GET, POST, PUT, DELETE	Error Message	Method not allowed on resource.

406 Not Acceptable	GET	Error Message	Requested representation not available for the resource.
408 Request Timeout	GET, POST	Error Message	Request has timed out.
409 Resource Conflict	PUT, POST, DELETE	Error Message	State of the resource doesn't permit request.
410 Gone	GET, PUT	Error Message	The URI used to refer to a resource.
411 Length Required	POST, PUT	Error Message	The server needs to know the size of the entity body and it should be specified in the Content Length header.
412 Precondition failed	GET	Error Message	Operation not completed because preconditions were not met.
413 Request Entity Too Large	POST, PUT	Error Message	The representation was too large for the server to handle.
414 Request URI too long	POST, PUT	Error Message	The URI has more than 2k characters.
415 Unsupported Type	POST, PUT	Error Message	Representation not supported for the resource.
416 Requested Range Not Satisfiable	GET	Error Message	Requested range not satisfiable.
500 Server Error	GET, POST, PUT	Error Message	Internal server error.
501 Not Implemented	POST, PUT, DELETE	Error Message	Requested HTTP operation not supported.
502 Bad Gateway	GET, POST, PUT, DELETE	Error Message	Backend service failure (data store failure).
505	GET	Error Message	HTTP version not supported.

Response Example



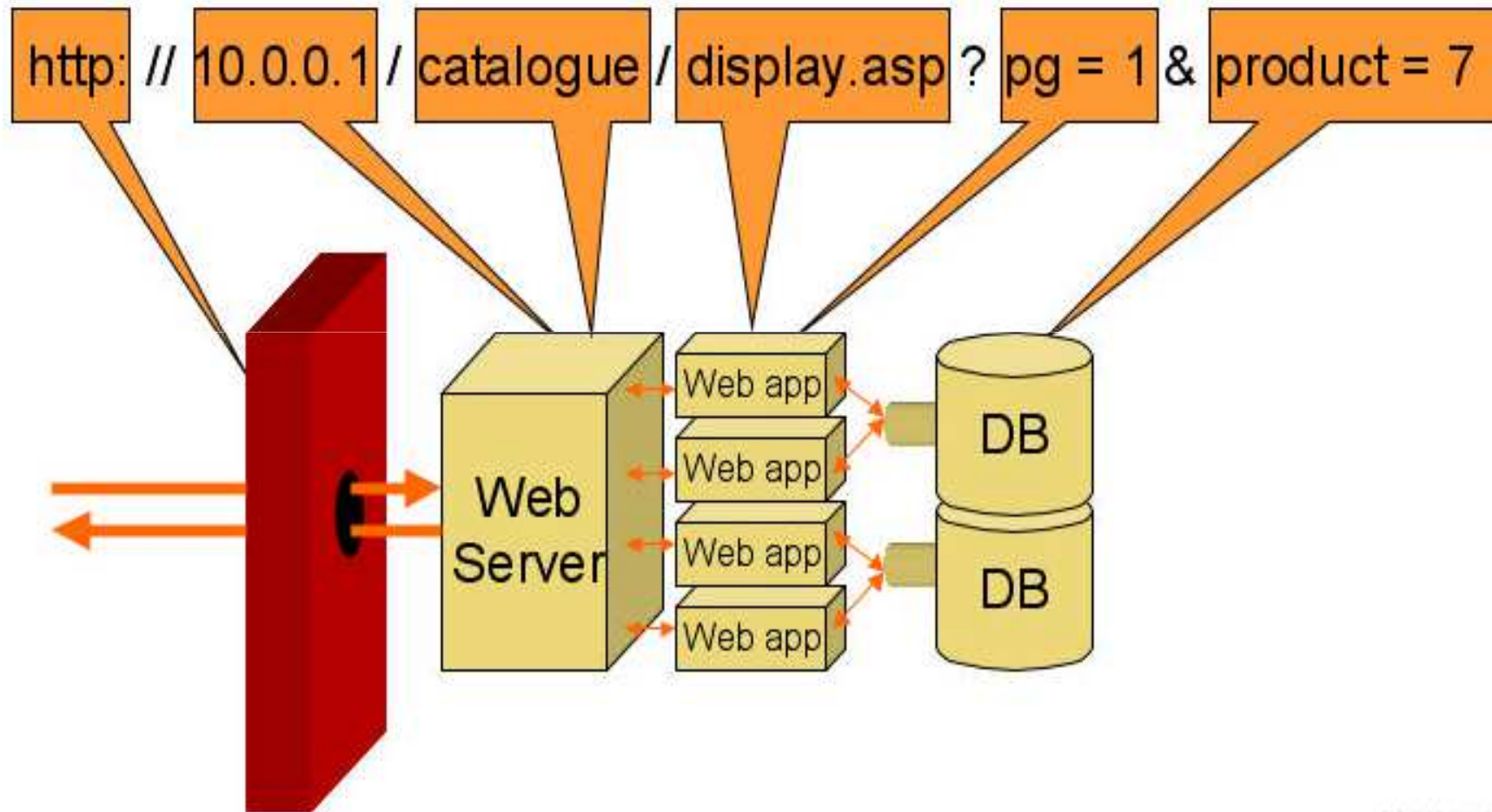
Web Server

- Software that accepts requests from clients (browsers, telnets) and then response that requests back to clients
- Example: Apache, Nginx, Xitami, Internet Information Services
- Using port to listen to (default:80)
- Using other configuration files
- Usually integrated with server scripting programming languages (PHP, ASP, JSP, etc)

Web Server basic capabilities

- **Virtual hosting:** multiple domain
- **Address mapping:** map request address to the actual file on server
- **Authentication:** by password, host credentials
- **Handling MIME** (multi purpose internet mail extensions)
- **Caching** support
- **Security** support

Mapping URL pada Web Server



Web Server Software

- The most popular Web server programs are:
 - Apache HTTP Server
 - Microsoft Internet Information Server (IIS)
- According to Netcraft
 - Accumulates popularity rankings

Developer	December 2010	Percent	January 2011	Percent	Change
Apache	661,722	66.61%	662,184	66.62%	0.01
Microsoft	163,589	16.47%	161,219	16.22%	-0.25
nginx	58,705	5.91%	60,782	6.12%	0.21
Google	19,623	1.98%	20,524	2.06%	0.09

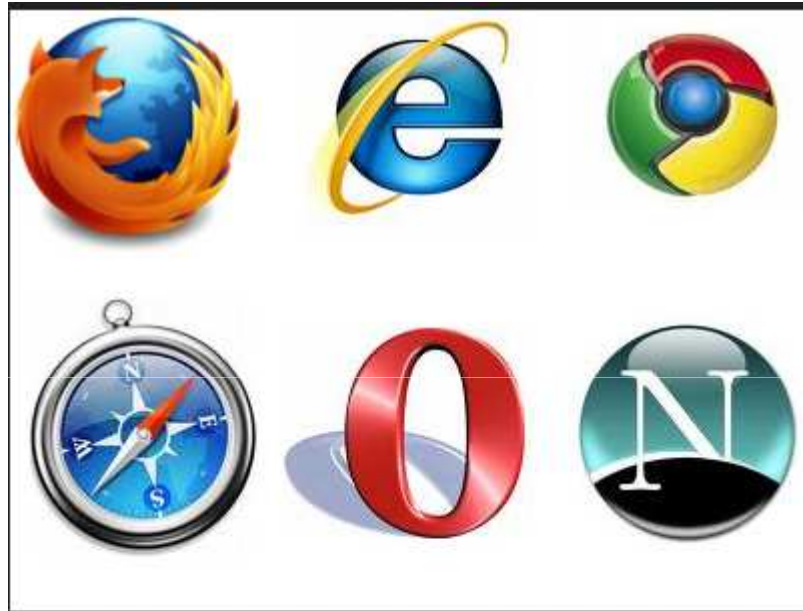
Web Server Environment Variables

<code>SERVER_PROTOCOL</code>	HTTP version as defined on the request line following HTTP method and URL.
<code>SERVER_PORT</code>	Server port used for submitting the request, set by the server based on the connection parameters.
<code>REQUEST_METHOD</code>	HTTP method as defined on the request line.
<code>PATH_INFO</code>	Extra path information in the URL. For example, if the URL is <code>http://mysite.org/cgi-bin/zip.cgi/test.html</code> , and <code>http://mysite.org/cgi-bin/zip.cgi</code> is the location of a CGI script, then <code>/test.html</code> is the extra path information.
<code>PATH_TRANSLATED</code>	Physical location of the CGI script on the server. In our example, it would be <code>/www/cgi-bin/zip.cgi</code> assuming that the server is configured to map the <code>/cgi-bin</code> path to the <code>/www/cgi-bin</code> directory.
<code>SCRIPT_NAME</code>	Set to the path portion of the URL, excluding the extra path information. In the same example, it's <code>/cgi-bin/zip.cgi</code>
<code>QUERY_STRING</code>	Information that follows the "?" in the URL.

Web Browsers

- All browsers are **clients**
- **Mosaic** - NCSA (Univ. of Illinois), in early 1993
 - First to use a GUI, led to explosion of Web use
 - Initially for X-Windows, under UNIX, but was ported to other platforms by late 1993
- Most **requests** are for existing **documents**, using HTTP
 - But some requests are for *program execution*, with the output being returned as a **document**

The Logos

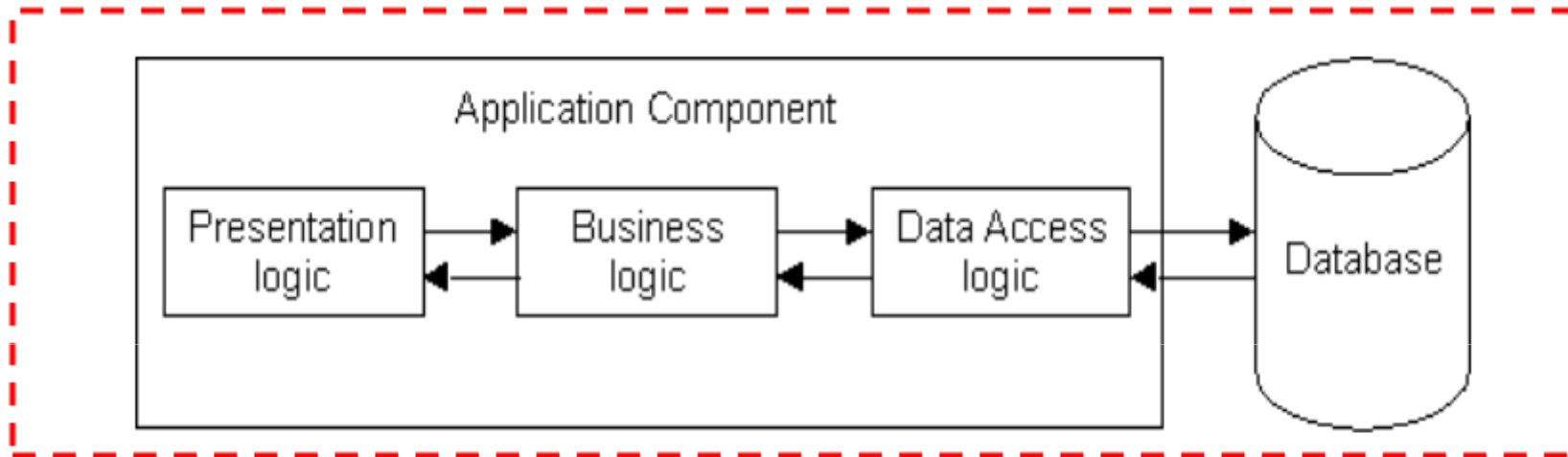


Web Tier

Web Architecture

- **Layering Aspect**
 - “Separation of concerns”
 - How many concurrent users are you serving?
 - Shared needs among multiple applications?
- **Data Aspect**
 - What kind(s) of data are you delivering?
 - Structured vs non structured
 - On-demand vs. real-time
 - What are the bandwidth requirements?
 - Size & nature of data
 - audience concerns

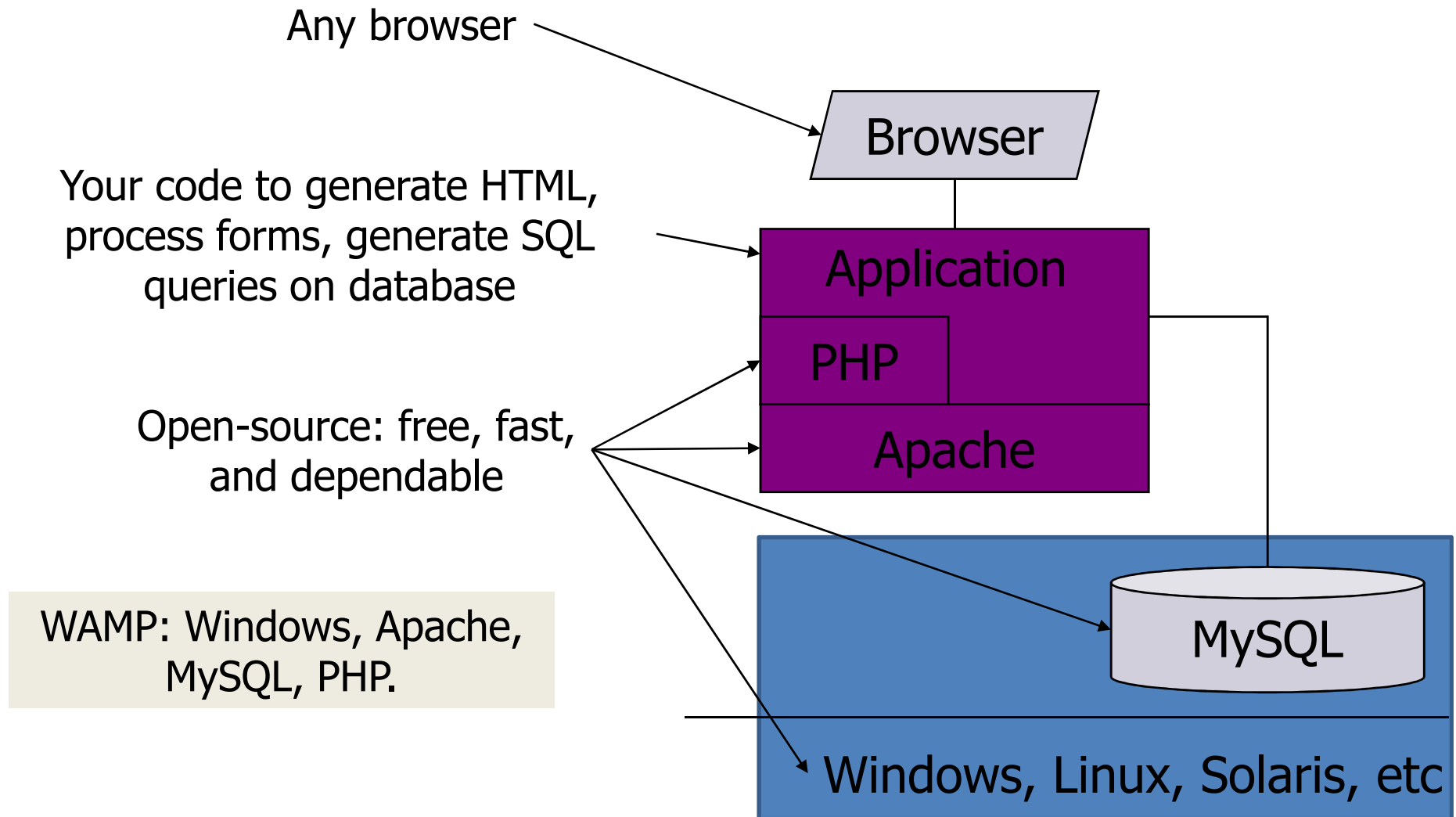
1-tier Architecture



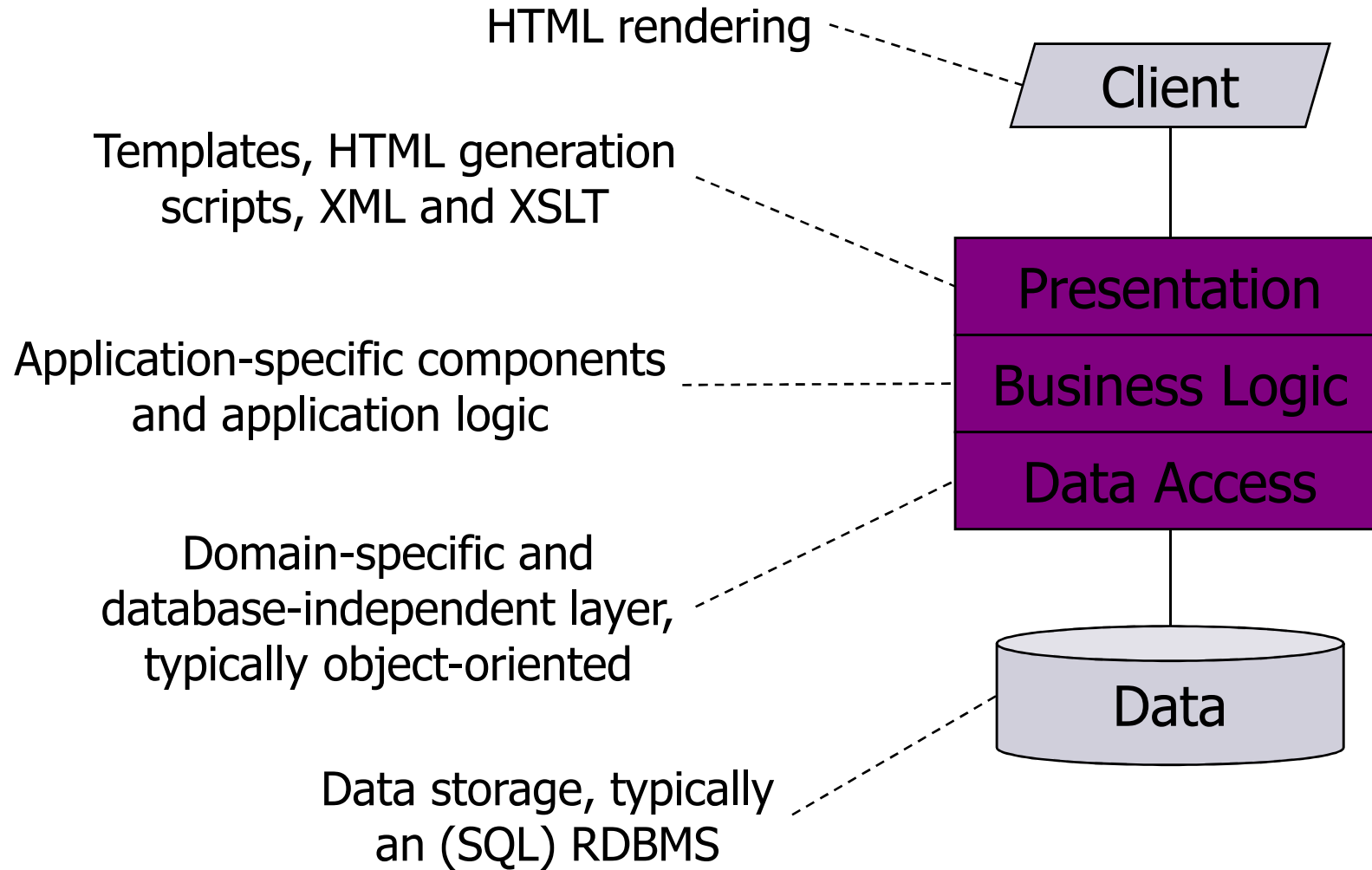
- All 3 layers are on the same machine
 - ▣ All code and processing kept on a single machine

Very easy way!

2-Tier Architecture



3-tier *web* architectures



The Tiers

- **Presentation Layer**

Static or dynamically generated content rendered by the browser (front-end)

- **Logic Layer**

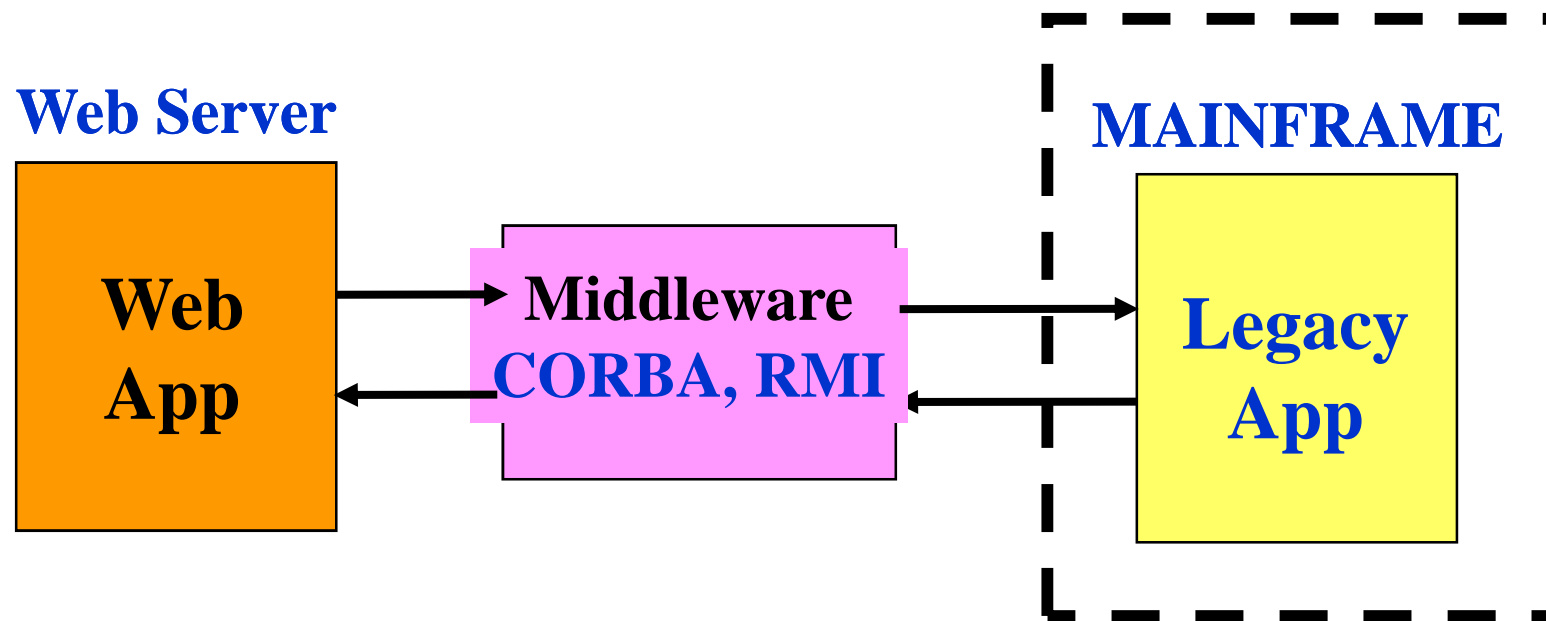
A dynamic content processing and generation level application server, e.g: Java EE, ASP.NET, PHP, ColdFusion platform

- **Data Layer**

A database, comprising both data sets and the database management system or RDBMS software that manages and provides access to the data (back-end)

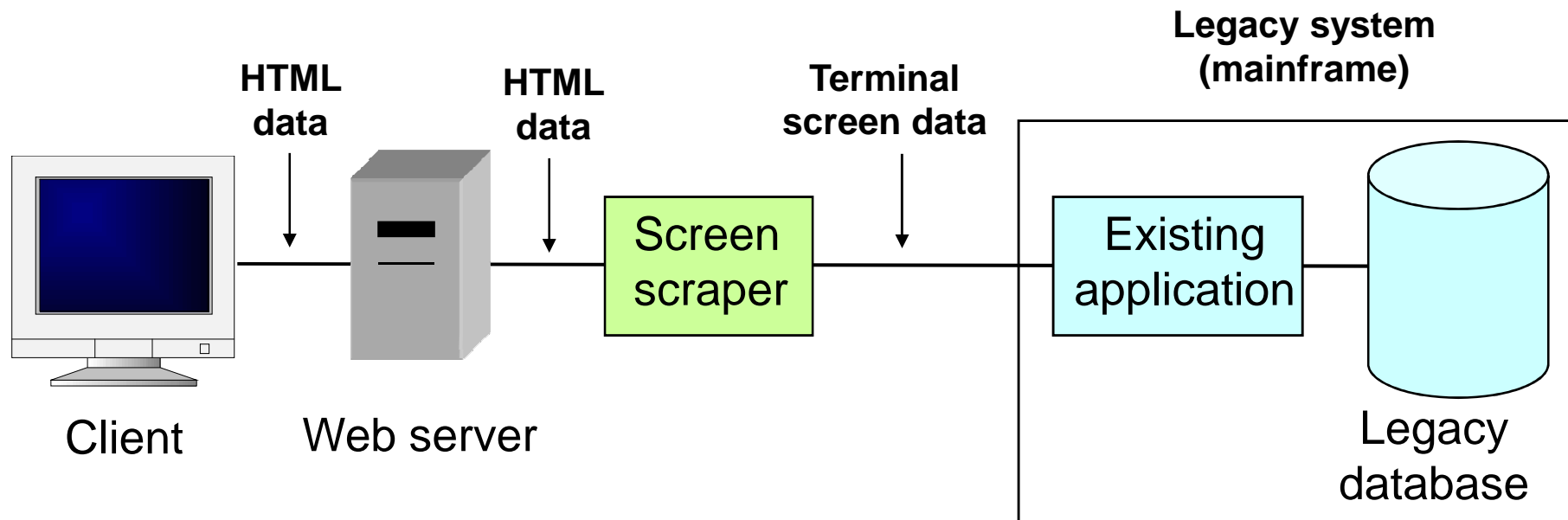
Web Integration Problem

Connecting to Legacy Systems Using middleware components



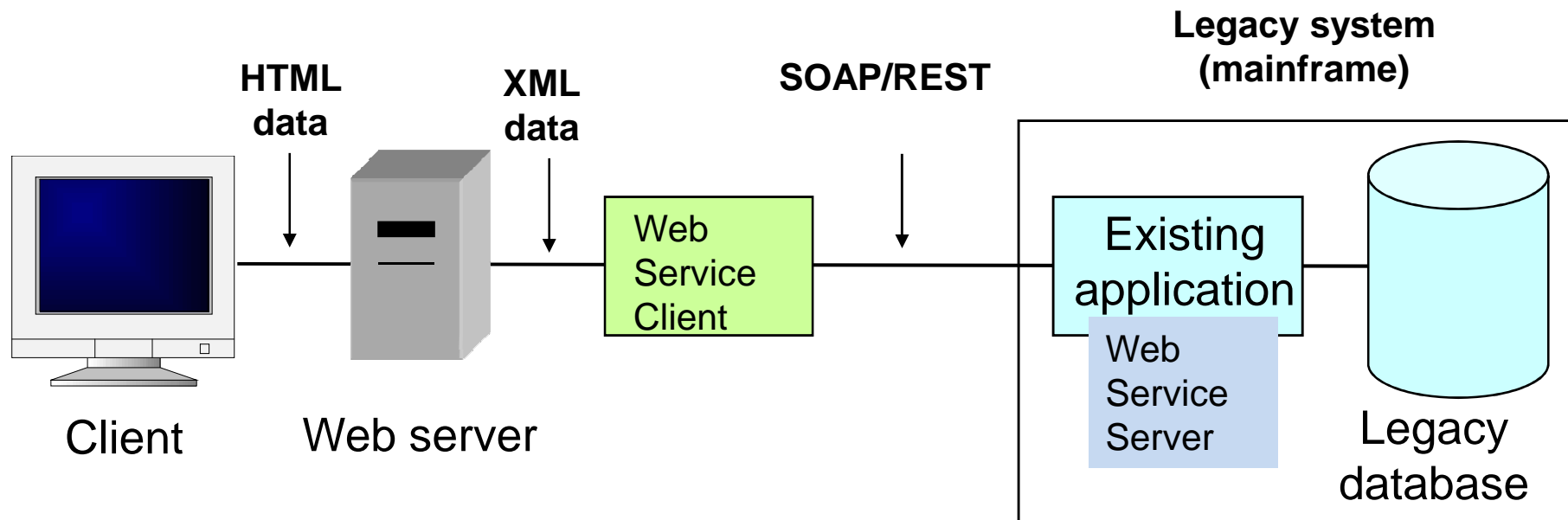
Connecting to Legacy Systems

The “screen scraper”



Connecting to Legacy Systems

The “web services”



Web Application Architecture

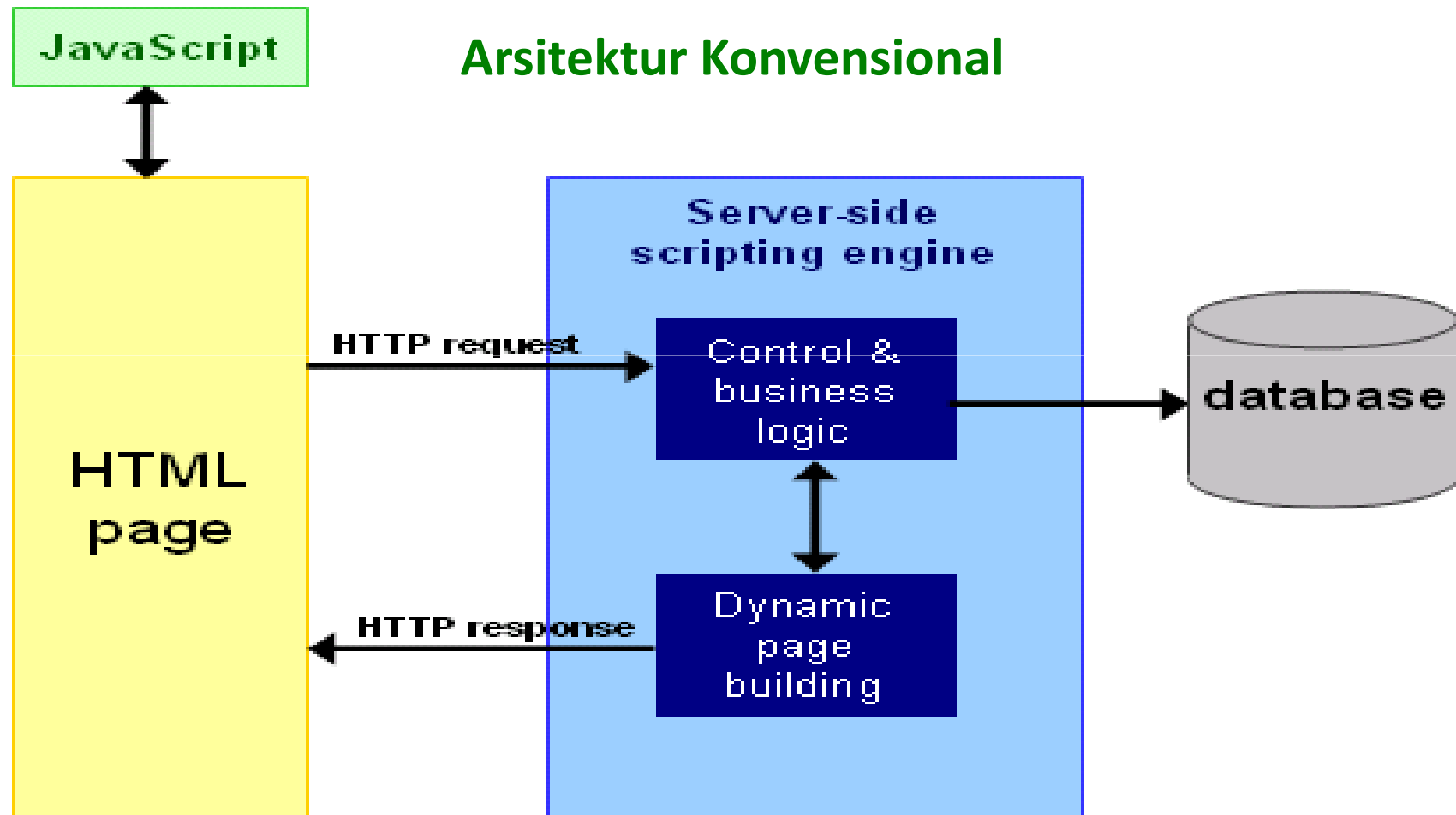
Server side

- Server hardware
- Web servers software
 - Apache, Xitami, IIS, GlassFish
- Server side programming language
 - PHP, JSP, ASP, ASP.NET, Ruby on Rails, Python, Perl, Cold Fusion
- MIME configuration
- Utility Programming Tools:
- Database Server
 - MySQL, SQLServer, Oracle

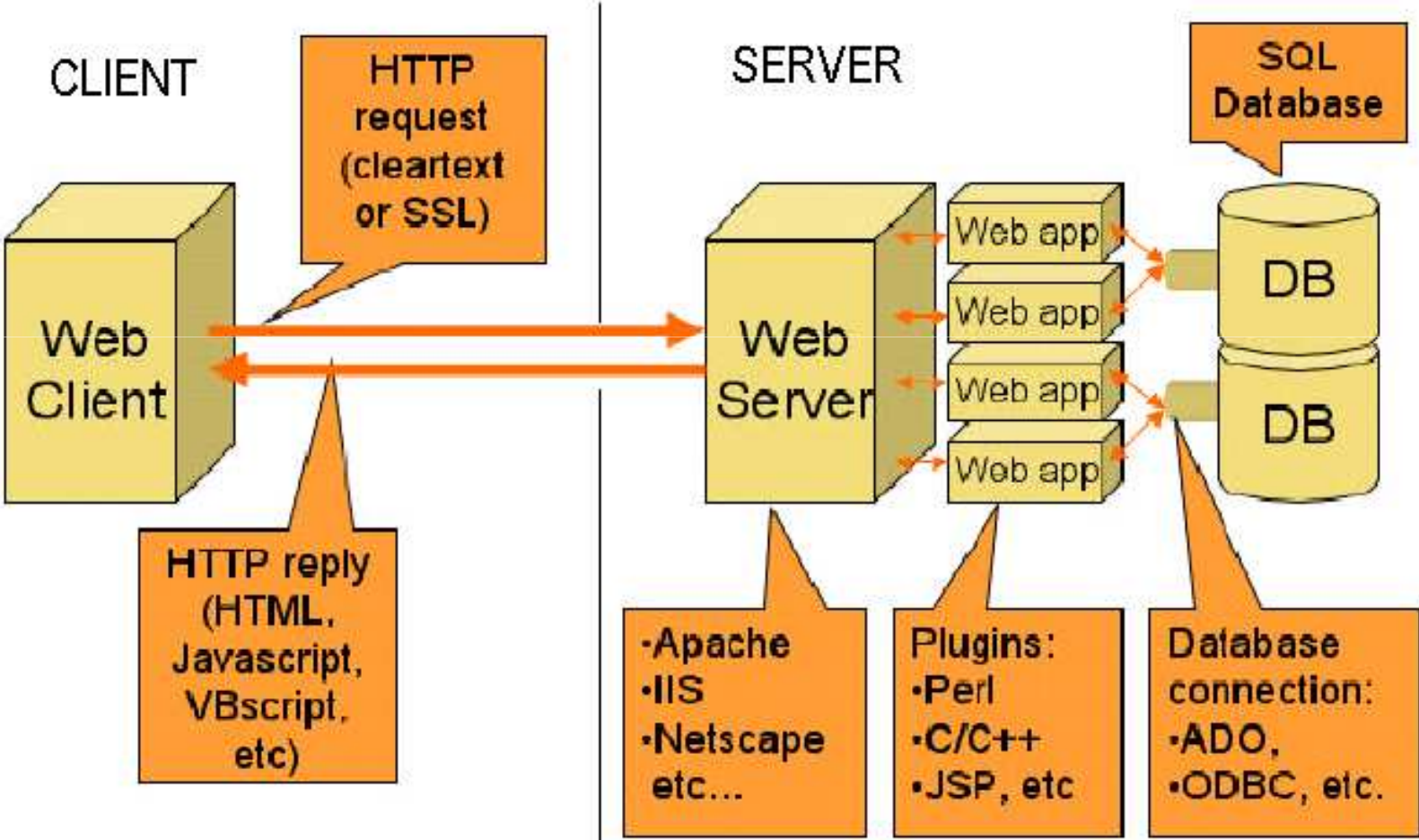
Client side

- Client hardware
- Web browsers
 - MIME config
- Language: HTML / XML / XHTML
- Client side scripting: Javascript / VBScript
- Design: CSS
- Interactive software:
 - Flash player
 - Java Applet
- Client software:
 - ActiveX / Plugin: program yg terintegrasi dgn browser
 - Helper: program yg terinstall di client

Arsitektur Aplikasi Web Konvensional

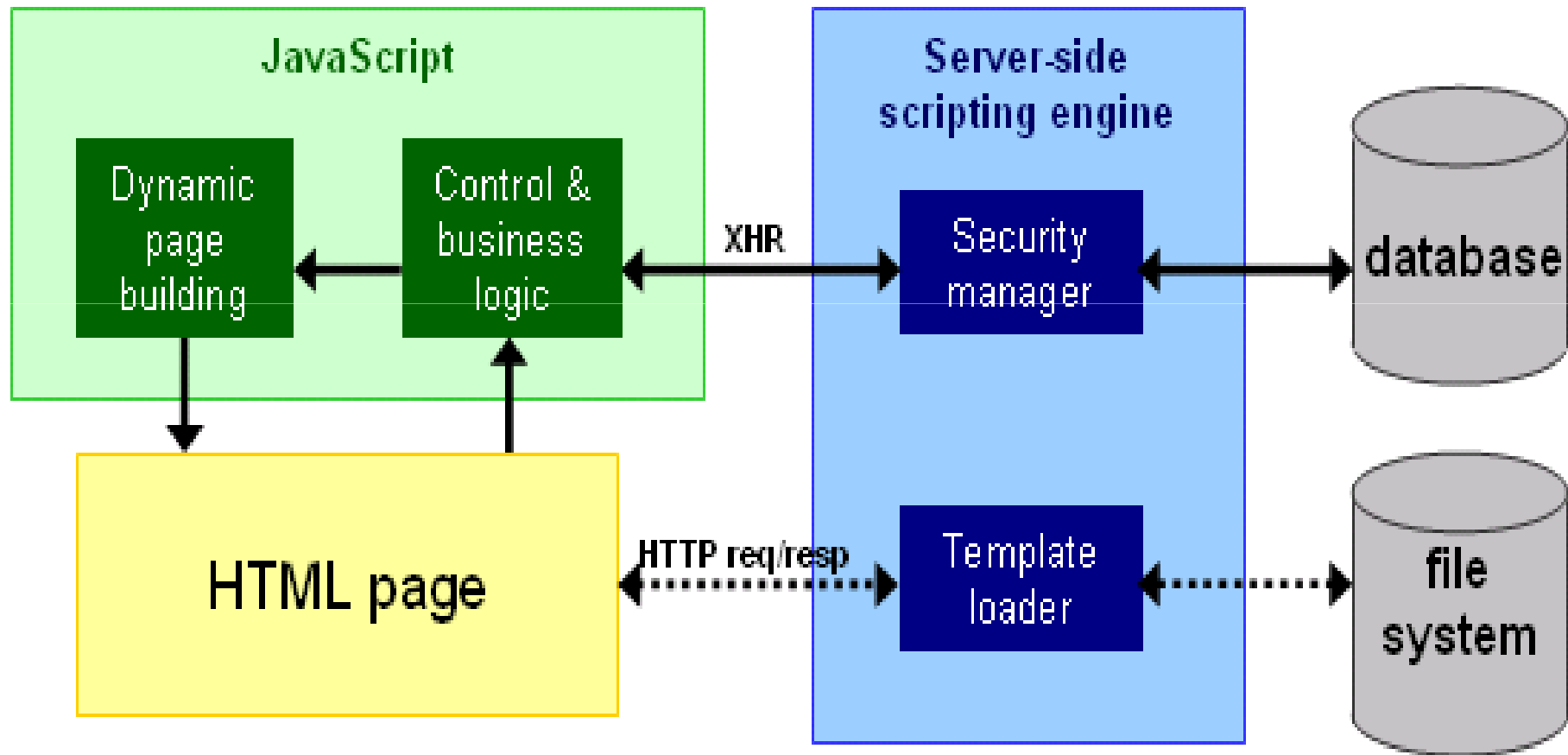


Client-Server Web Detail



Arsitektur Aplikasi Web Sekarang

Tren Sekarang (Ajax-Based)



Web Data Aspect architecture

Data aspect architecture

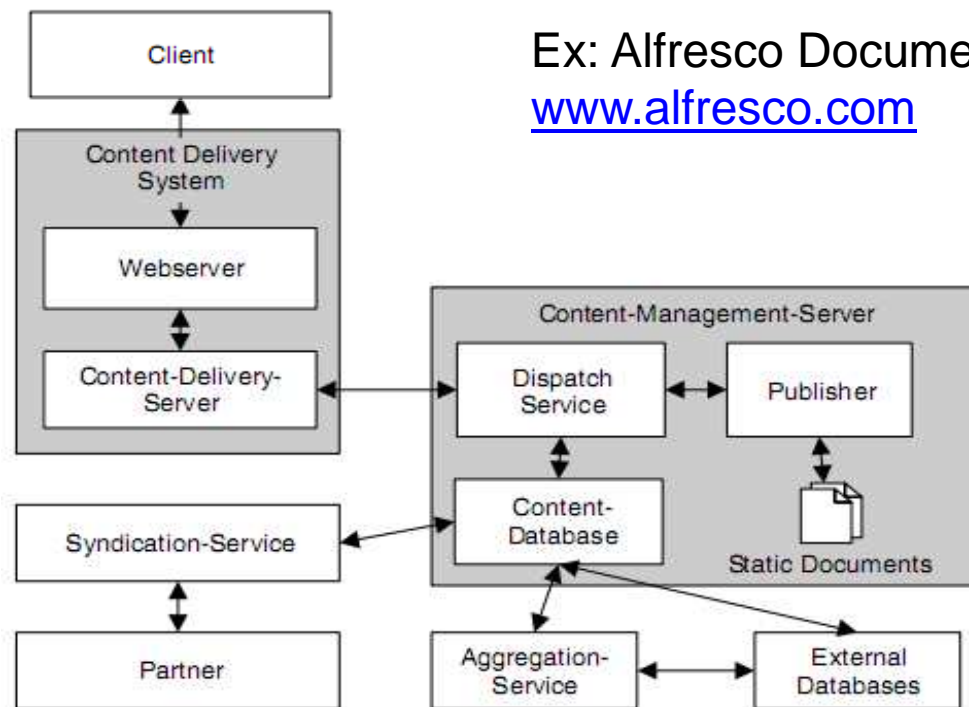
- **Structured data** of the kind held in **database**
- **Documents** of the kind used in **document management systems**
- **Multimedia data** of kind held in **media servers** and **multimedia management systems**

Database centric architecture

- **Integrate DB into Web app**
- DB are accessed either directly from within Web server extension, such as:
 - JDBC
 - ODBC
 - ADO
 - ADO.NET

Architecture of Document Management Systems

- **Document management systems**
 - Integrate documents from **different sources**



A content management architecture for Web applications (according to Anastopoulos and

DMS Capabilities



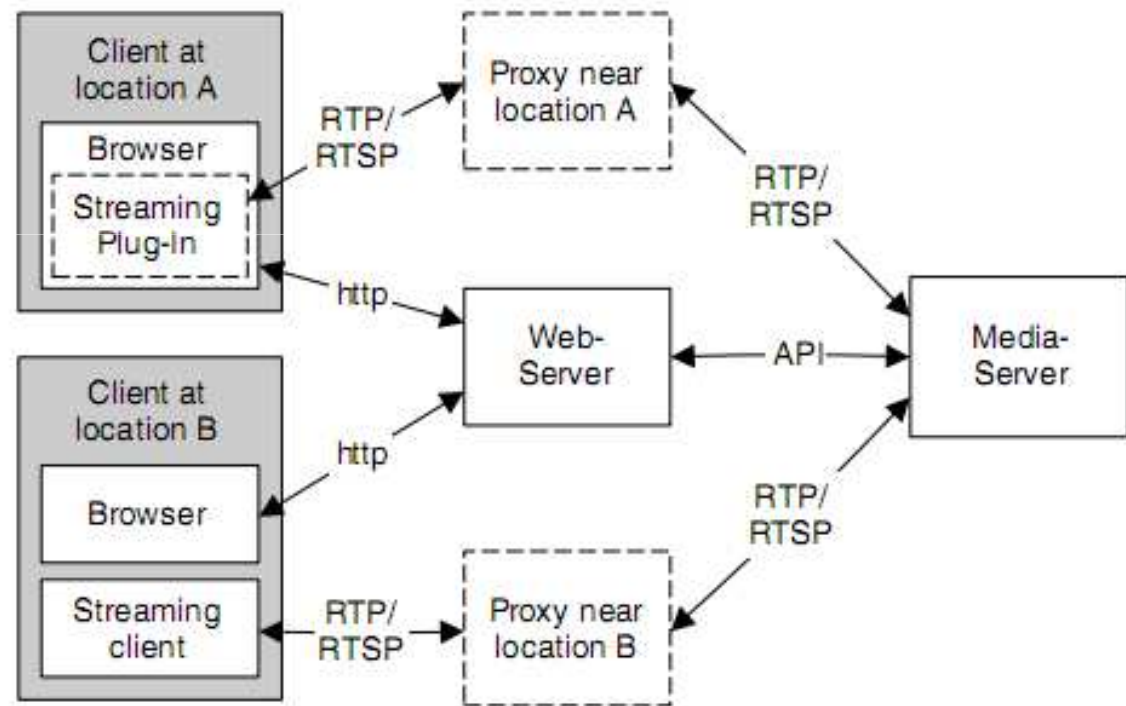
Architecture of multimedia data

- **Streaming**

- Client can begin play out of audio few second after it begins receiving the file from server
- Corresponding bandwidth, low jitter
- Real time protocol

- **Example:**

- Application **Helix Media Server**
- Web: **Youtube**



Streaming media architecture using point-to-point connections.

Web Application

Web App

Case Study : Yahoo! Directory

Provider-defined directory

Yahoo! My Yahoo! Mail Welcome, **Guest** [Sign In] [Directory Home](#) [Help](#)

YAHOO! DIRECTORY Search: the Web | the Directory

Yahoo! Directory [Advanced Search](#) [Suggest a Site](#) [Email This Page](#)

Arts & Humanities
[Photography](#), [History](#), [Literature](#)...

Business & Economy
[B2B](#), [Finance](#), [Shopping](#), [Jobs](#)...

Computers & Internet
[Hardware](#), [Software](#), [Web](#), [Games](#)...

Education
[Colleges](#), [K-12](#), [Distance Learning](#)...

Entertainment
[Movies](#), [TV Shows](#), [Music](#), [Humor](#)...

Government
[Elections](#), [Military](#), [Law](#), [Taxes](#)...

Health
[Diseases](#), [Drugs](#), [Fitness](#), [Nutrition](#)...

News & Media
[Newspapers](#), [Radio](#), [Weather](#), [Blogs](#)...


Recreation & Sports
[Sports](#), [Travel](#), [Autos](#), [Outdoors](#)...

Reference
[Phone Numbers](#), [Dictionaries](#), [Quotes](#)


The Spark: The Meerkat in the Marketing Department



By Liz Gill
Fri, July 10, 2009, 12:01 am PDT

In company logos and advertisements, animals are a fairly common sight. So common, in fact, that you probably haven't given these mascots a great deal of thought. But those whose job involves selecting an animal representative for an organization or company face no small task. With their choice, they will be aligning themselves with whatever qualities people tend to associate with that animal. Once they've found their totem, they may forever be associated with it.



"His Master's Voice"
(What you don't see is that Nipper is sitting on the coffin of his dead owner)

A prime example of logo longevity is "His Master's Voice," a design trademarked by the Victor Record Company on this day in 1900, sealing forever in our minds the image of Nipper the dog listening attentively to a Victrola, and connecting it further with the Gramophone Company (later known as HMV). In the ensuing century, plenty of other corporate critters have made their marks on our cultural consciousness. How many times have you heard Leo, the  MGM lion roar?

Financial institutions have adopted symbols of strength, including an elk and a lion, though Merrill Lynch was wise to select a bull rather than a bear. Animals in advertising have also provided  memorable and  humorous moments.

Ads by Yahoo!

[Record Players](#)
Browse a huge selection now. Find exactly what you want today.
www.ebay.com

[Tyson Animal Hospital](#)
Read reviews for this Pet & Animal Service & find Pet & Animal Info.
Durham.Citysearch.com

[Animal / Critter Control](#)
Local Animal Control experts Call us to remove nuisance animals.
www.crittercontrol.com

[Crosley Record Players](#)
Don't miss this Crosley Sale
www.McGalls.com

[Animal House T-Shirts](#)
Stylish, exceptional Animal House t-shirts.
80sTees.com

Web App

Case Study : Facebook

The image shows a screenshot of the Facebook web interface with several annotations in cyan boxes and arrows pointing to specific features:

- Friends**: Points to the 'Friends' link in the left-hand navigation menu.
- photos**: Points to the 'Photos' link in the left-hand navigation menu.
- Video**: Points to the 'Video' link in the left-hand navigation menu.
- Aggregation with Picasa**: Points to the 'Picasa' link in the left-hand navigation menu.
- More apps!**: Points to the 'Applications' link in the bottom navigation bar.
- groups**: Points to the 'University Information Technology Services' group post at the bottom of the news feed.
- Your current status**: Points to the 'What's on your mind?' text input field at the top of the news feed.
- Activities of your friends**: A bracketed annotation pointing to the posts by Rui Wang, Joel Cooper, and Huijun Wang.
- Comment, Rate**: Points to the 'Comment' and 'Like' buttons on Huijun Wang's post.
- Chat**: Points to the 'Chat (2)' button in the bottom right corner.

Google Web Applications

- Google Docs
- Google Search Engine
- Google Blog (Blogspot)
- Google Books
- Google Play (Android)
- Google Translate
- Google Scholar
- Google Video (YouTube)
- Google AdSense, AdWord
- Google Calendar
- Google Groups
- Google Earth
- Google Map
- Google Drive
- Google Hangout
- Google+
- Picasa
- Go.gl
- Google Analytics

Examples of two “versions” of web apps

1995-2005 Web	2005-Present Web
Britannica Online	Wikipedia
Akamai	BitTorrent
Directories (Taxonomy)	Tagging (Folksonomy)
Tightly coupled apps	App Mashup/Integration
Home page	Blog

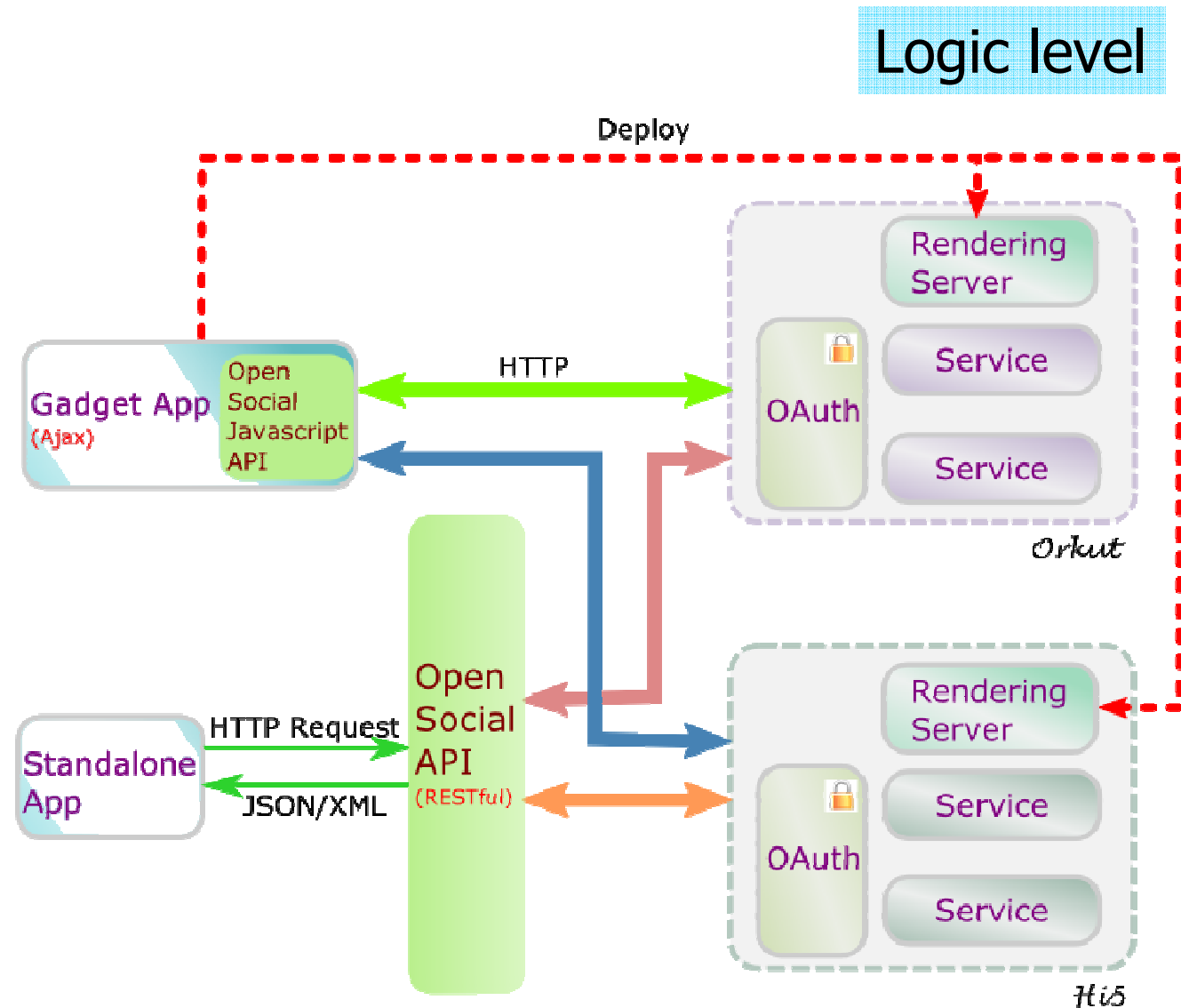
OpenSocial

- A coherent open architecture designed for **social network** services and applications.
 - Common APIs across many websites
 - REST/RPC protocols – for server-to-server interactions
 - Javascript APIs – for browser-to-server interactions
 - Authorization mechanism, Data model ...
- **Usage**
 - Supported by MySpace, Google Orkut, Twitter, LinkedIn, XiaoNei...
 - Internationalization
 - Rival: Facebook

OpenSocial – Architecture example

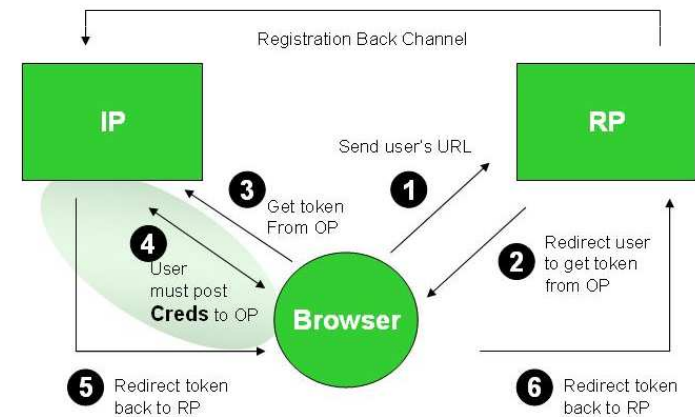
Components

- Interface – REST, Javascript APIs
- Client – Ajax, Gadget
- Message Format – JSON, XML
- Security - OAuth
- Data Model



Authentication – OpenID

- Motivation
 - Provide lightweight **authentication** service across domains
- Solution
 - Users are asked to prove ownership of their OpenID identifiers.
 - OpenID identifiers are URLs
 - Authentication delegation (service provider → identity provider)
- Advantages
 - Cross-domain authentication
 - Attribute exchange beyond authentication
 - Single Sign-On
 - Easy OpenID provider switch
- Drawbacks
 - Phishing attack
- Resources
 - Supported by Facebook, Verisign, Sourceforge, Yahoo, etc.



Next

- Service Oriented Architectures