Administrarea Bazelor de Date
Managementul în Tehnologia Informaţiei

Sisteme Informatice şi Standarde Deschise (SISD)

2009-2010

Curs 4

Standarde Web

Inside the Web

- Main protocols
  - URL, HTML, and HTTP
  - HTTP: the protocol and its stateless property
- Web Systems Components
  - Clients
  - Servers
  - DNS (Domain Name System)
- Interaction with underlying network protocol: TCP
- Scalability and performance enhancement
  - Server farms
  - Web Proxy
  - Content Distribution Network (CDN)

Web History

- Before the 1970s-1980s
  - CERN Initiative
  - Internet used mainly by researchers and academics
  - Log in remote machines, transfer files, exchange e-mail
- Internet growth and commercialization
  - 1988: ARPANET gradually replaced by the NSFNET
  - Early 1990s: NSFNET begins to allow commercial traffic
- Initial proposal for the Web by Berners-Lee in 1989
- Enablers for the success of the Web
  - 1980s: Home computers with graphical user interfaces
  - 1990s: Power of PCs increases, and cost decreases

Main ingredients of the Web

- **URL**
  - Denotes the global unique location of the web resource
  - Formatted string

    e.g., http://www.princeton.edu/index.html

    Protocol for communicating with server (e.g., http)

    Name of the server (e.g., www.princeton.edu)

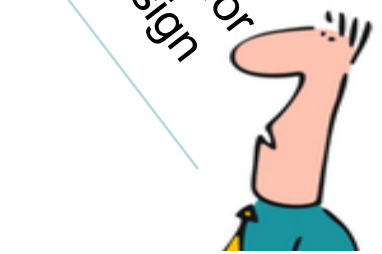
    Name of the resource (e.g., index.html)
- **HTML (HyperText Markup Language)**
  - Format text, reference images, embed hyperlinks
  - Representation of hypertext documents in ASCII format
  - Interpreted by Web browsers when rendering a page
  - Straight-forward and easy to learn
  - Simplest HTML document is a plain text file
  - Automatically generated by authoring programs

Main ingredients of the Web

- **Web page**
  - Base HTML file
  - referenced objects (e.g., images), Each object has its own URL
- **HTTP**
  - Protocol for client/server communication
- **Client program**
  - E.g., Web browser
  - Running on end host
- **Server program**
  - E.g., Web server
  - Provides service

HTTP
Request Methods and Response Codes

- Request methods include
  - GET: return current value of resource, …
  - HEAD: return the meta-data associated with a resource
  - POST: update a resource, provide input to a program, …
  - Etc.
- Response code classes
  - 1xx: informational (e.g., “100 Continue”)
  - 2xx: success (e.g., “200 OK”)
  - 3xx: redirection (e.g., “304 Not Modified”)
  - 4xx: client error (e.g., “404 Not Found”)
  - 5xx: server error (e.g., “503 Service Unavailable”)



HTTP is a Stateless Protocol

- Stateless
  - Each request-response exchange treated independently
  - Clients and servers not required to retain state
- Statelessness to improve scalability
  - Avoids need for the server to retain info across requests
  - Enables the server to handle a higher rate of requests

Web Systems Components

- **Clients**
  - Send requests and receive responses
  - Browsers, spiders, and agents
- **Servers**
  - Receive requests and send responses
  - Store or generate the responses
- **DNS (Domain Name System)**
  - Distributed network infrastructure
  - Transforms site name -> IP address
  - Direct clients to servers

Web Browser

- Generating HTTP requests
  - User types URL, clicks a hyperlink, or selects bookmark
  - User clicks “reload”, or “submit” on a Web page
  - Automatic downloading of embedded images
- Layout of response
  - Parsing HTML and rendering the Web page
  - Invoking helper applications (e.g., Acrobat, PowerPoint)
- Maintaining a cache
  - Storing recently-viewed objects
  - Checking that cached objects are fresh

Typical Web Transaction

- User clicks on a hyperlink
  - http://florinpop.ro/index.html
- Browser learns the IP address of the server
  - Invokes `gethostbyname(florinpop.ro)` and gets a return value of 141.85.99.132
- Browser establishes a TCP connection
  - Selects an ephemeral port for its end of the connection
  - Contacts 141.85.99.132 on port 80
- Browser sends the HTTP request
  - "GET /index.html HTTP/1.1
    Host: florinpop.ro"

Typical Web Transaction

- Browser parses the HTTP response message
  - Extract the URL for each embedded image
  - Create new TCP connections and send new requests
  - Render the Web page, including the images
- Opportunities for caching in the browser
  - HTML file
  - Each embedded image
  - IP address of the Web site

Web Server

- Web site vs. Web server
  - Web site: collections of Web pages associated with a particular host name
  - Web server: program that satisfies client requests for Web resources
- Handling a client request
  - Accept the TCP connection
  - Read and parse the HTTP request message
  - Translate the URL to a filename
  - Determine whether the request is authorized
  - Generate and transmit the response

Web Server: Generating a Response

- Returning a file
  - URL corresponds to a file (e.g., /public_html/index.html)
  - … and the server returns the file as the response
  - … along with the HTTP response header
- Returning meta-data with no body
  - Example: client requests object “if-modified-since”
  - Server checks if the object has been modified
  - … and simply returns a “HTTP/1.1 304 Not Modified”
- Dynamically-generated responses
  - URL corresponds to a program the server needs to run
  - Server runs the program and sends the output to client

Hosting: Multiple Sites Per Machine

- Multiple Web sites on a single machine
  - Hosting company runs the Web server on behalf of multiple sites (e.g., florinpop.ro and diogenes.grid.pub.ro)
- Problem: returning the correct content
  - florinpop.ro/index.html vs. diogenes.grid.pub.ro/index.html
  - How to differentiate when both are on same machine?
- Solution: multiple servers on the same machine
  - Run multiple Web servers on the machine
  - Have a separate IP address for each server

Hosting: Multiple Machines Per Site

- Replicating a popular Web site
  - Running on multiple machines to handle the load
  - … and to place content closer to the clients
- Problem: directing client to a particular replica
  - To balance load across the server replicas
  - To pair clients with nearby servers
- Solution:
  - Takes advantage of Domain Name System (DNS)

DNS Query in Web Download

- User types or clicks on a URL
  - E.g., florinpop.ro/phd/index.html
- Browser extracts the site name
  - E.g., florinpop.ro
- Browser calls gethostbyname() to learn IP address
  - Triggers resolver code to query the local DNS server
- Eventually, the resolver gets a reply
  - Resolver returns the IP address to the browser
- Then, the browser contacts the Web server
  - Creates and connects socket, and sends HTTP request

Clever Load Balancing Schemes

- Selecting the “best” IP address to return
  - Based on server performance
  - Based on geographic proximity
  - Based on network load
  - …
- Example policies
  - Round-robin scheduling to balance server load
  - U.S. queries get one address, Europe another
  - Tracking the current load on each of the replicas

TCP Interaction: Multiple Transfers

- Most Web pages have multiple objects (HTML file and multiple embedded images)
- Serializing the transfers is not efficient
- Parallel connections
  - Browser opens multiple TCP connections
  - … and retrieves a single image on each connection
- Performance trade-offs
  - Multiple downloads sharing the same network links
  - Unfairness to other traffic traversing the links
- Most HTTP transfers are short
  - Very small request message (e.g., a few hundred bytes)
  - Small response message (e.g., a few kilobytes)
- TCP overhead may be big
  - Three-way handshake to establish connection
  - Four-way handshake to tear down the connection

TCP Interaction: Short Transfers

- Round-trip time estimation
  - Maybe large at the start of a connection (e.g., 3 seconds)
  - Leads to latency in detecting lost packets
- Congestion window
  - Small value at beginning of connection
  - May not reach a high value before transfer is done
- Detecting packet loss
  - Timeout: slow ☹
  - duplicate ACK

TCP Interaction: Persistent Connections

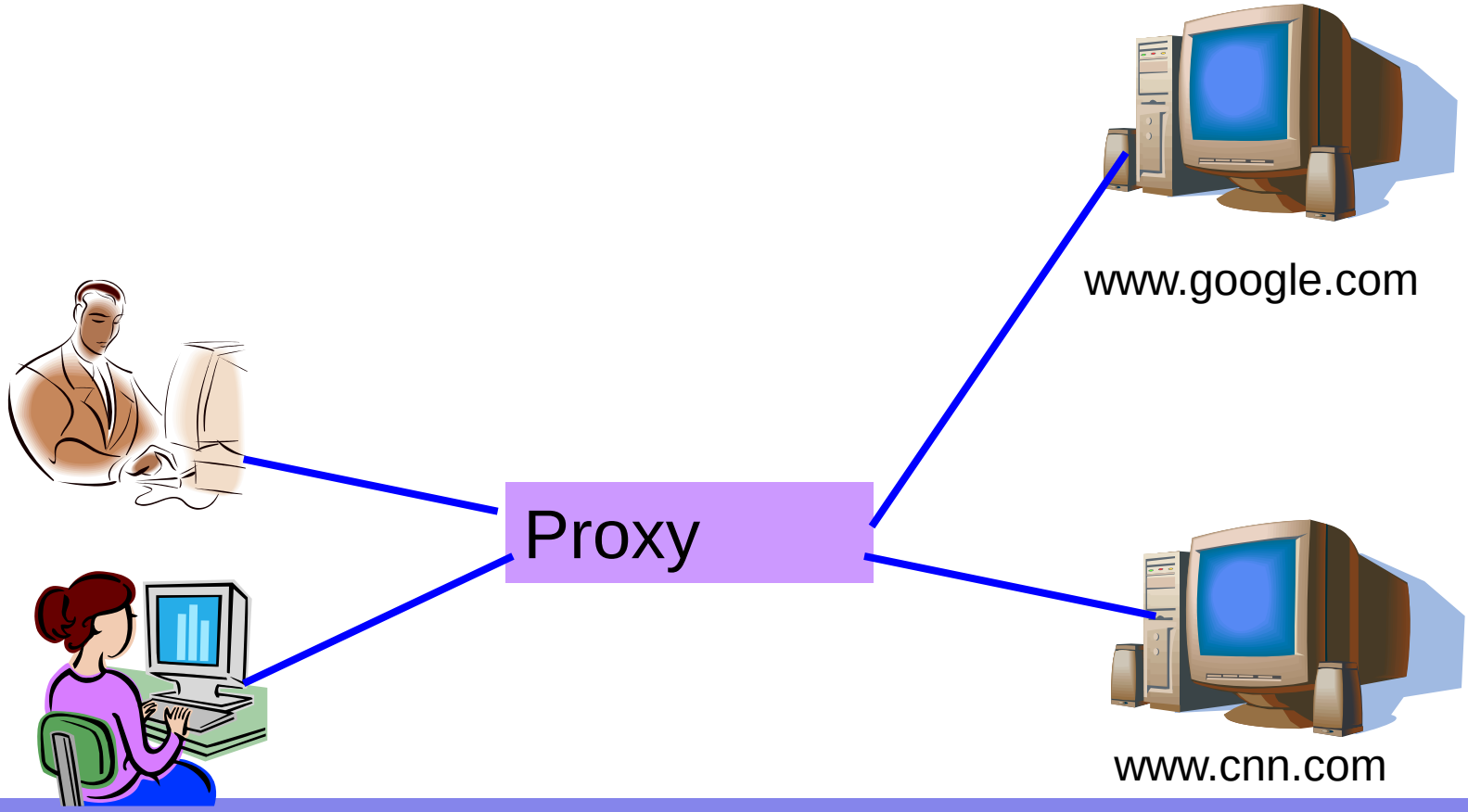
- Handle multiple transfers per connection
  - Maintain the TCP connection across multiple requests
  - Either the client or server can tear down the connection
  - Added to HTTP after the Web became very popular
- Performance advantages
  - Avoid overhead of connection set-up and tear-down
  - Allow TCP to learn a more accurate RTT estimate
  - Allow the TCP congestion window to increase

Server Farms

- Definition
  - a collection of computer servers to accomplish server needs far beyond the capacity of one machine.
  - Often have both a primary and backup server allocated to a single task (for fault tolerance)
- Web Farms
  - Common use of server farms is for web hosting

Web Proxies are Intermediaries

- Proxies play both roles
  - A server to the client
  - A client to the server



Proxy Caching

- Client #1 requests http://www.foo.com/fun.jpg
  - Client sends “GET fun.jpg” to the proxy
  - Proxy sends “GET fun.jpg” to the server
  - Server sends response to the proxy
  - Proxy stores the response, and forwards to client
- Client #2 requests http://www.foo.com/fun.jpg
  - Client sends “GET fun.jpg” to the proxy
  - Proxy sends response to the client from the cache
- Benefits
  - Faster response time to the clients
  - Lower load on the Web server
  - Reduced bandwidth consumption inside the network

Getting Requests to the Proxy

- Explicit configuration
  - Browser configured to use a proxy
  - Directs all requests through the proxy
  - Problem: requires user action
- Transparent proxy (or “interception proxy”)
  - Proxy lies in path from the client to the servers
  - Proxy intercepts packets en route to the server
  - … and interposes itself in the data transfer
  - Benefit: does not require user action

Other Functions of Web Proxies

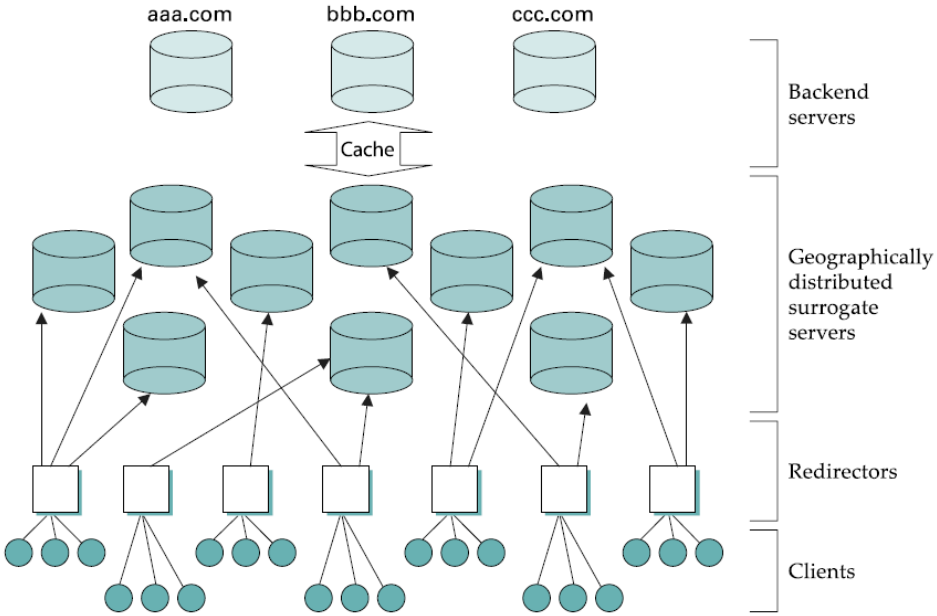
- Anonymization
  - Server sees requests coming from the proxy address
  - … rather than the individual user IP addresses
- Transcoding
  - Converting data from one form to another
  - E.g., reducing the size of images for cell-phone browsers
- Prefetching
  - Requesting content before the user asks for it
- Filtering
  - Blocking access to sites, based on URL or content

Motivation for Content Distribution Network (CDN)

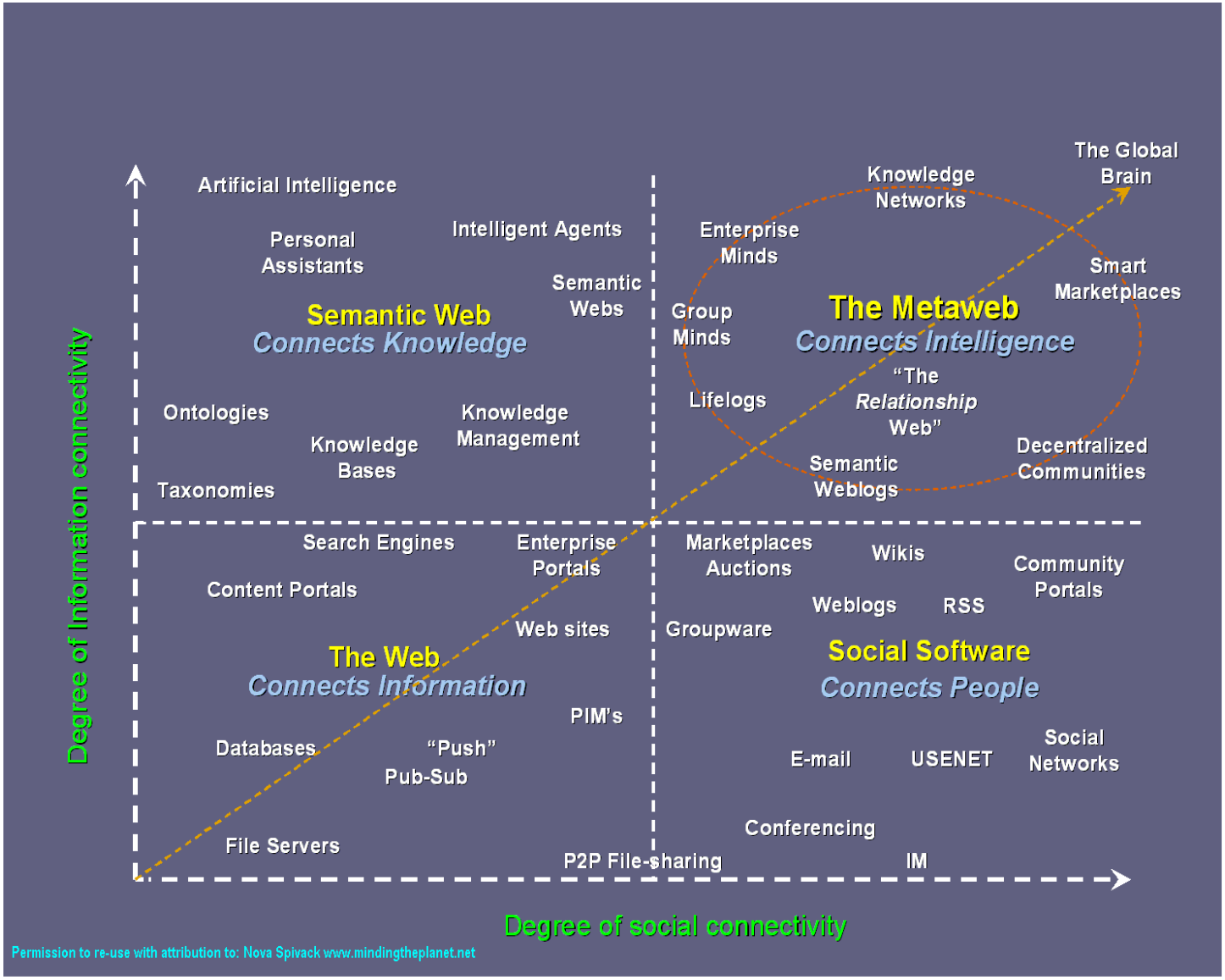
- Providers want to offer content to consumers
  - Efficiently
  - Reliably
  - Securely
  - Inexpensively
- The server and its link can be overloaded
- Peering points between ISPs can be congested
- Alternative solution: Content Distribution Networks
  - Geographically diverse servers serving content from many sources
- Proactively replicate data by caching static pages

CDN Architecture

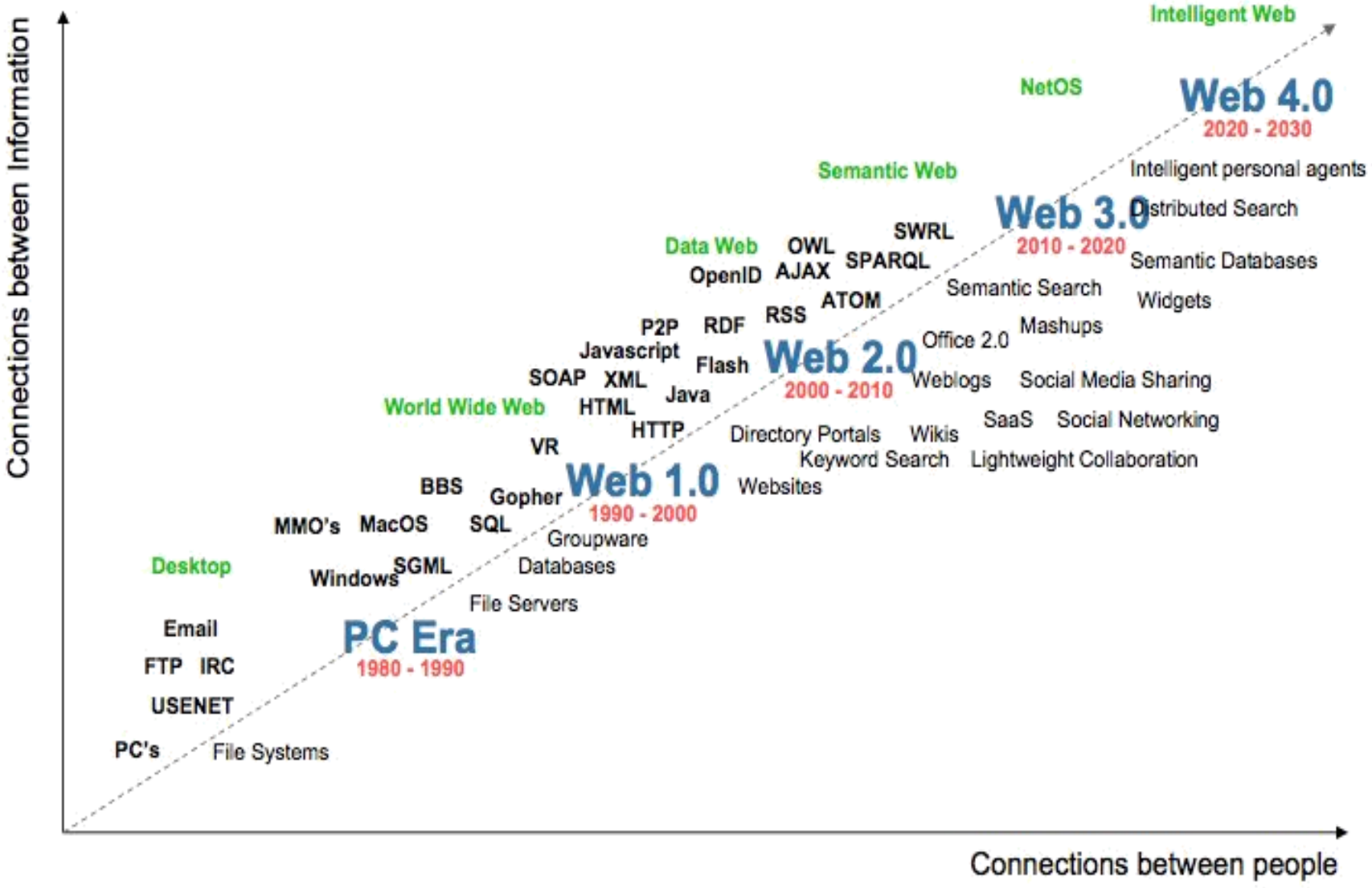
- Architecture
  - Backend servers
  - Geographically distributed surrogate servers
  - Redirectors (according to network proximity, balancing)
  - Clients
- Redirector Mechanisms
  - Augment DNS to return different server addresses
  - Server-based redirection: based on HTTP redirect feature



Evolution of Web Services



Evolution of Web Services



Exam's quizzes

- **1.** Descrieţi pe scurt principalele componente ale Web-ului. Care sunt principalele probleme în arhitecturile Web? Care sunt principalele soluții la aceste probleme?
- **2.** Descrieți modul de interacțiune pentru modelul client-server în sistemul Web.
- **3.** Care sunt principalele utilizări ale protocolului TCP în Web?
- **4.** Ce înțelegeți prin Web Proxy? Care sunt principalele caracteristici?
- **5.** Descrieți arhitectura CDN și avantajele utilizării.