Madrid, February 2007

Distributed Web-based architectures for content and service delivery

Michele Colajanni
Department of Information Engineering
University of Modena and Reggio Emilia
colajanni@unimo.it

WEBlab Group

- **Web Engineering and Benchmarking Laboratory**
- **Founded in Roma in 1996, then in Modena from 1998 (Dept. of Information Engineering)**

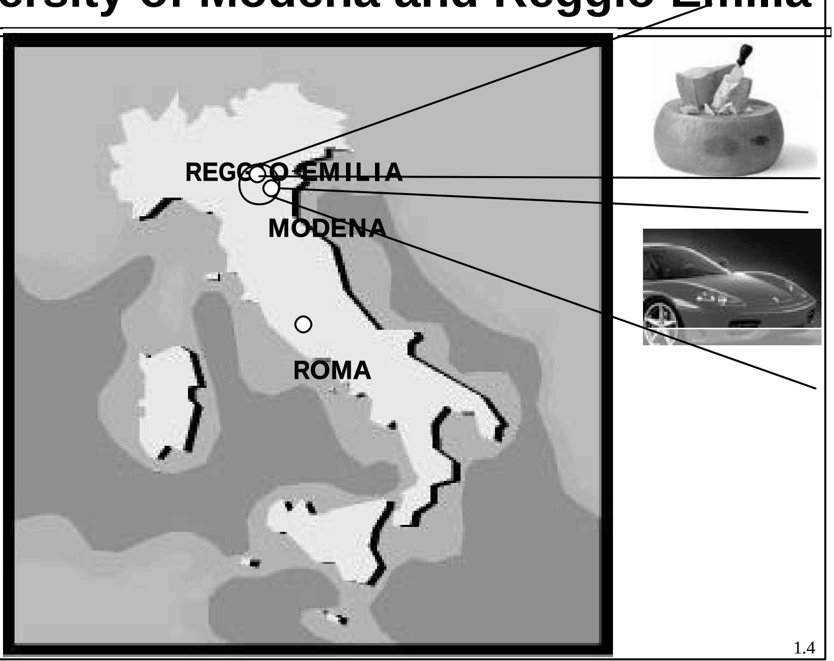
- **Main research topics:**
  - Infrastructures for the Web
  - Parallel and distributed systems
  - Performance
  - Security
  - Infrastructures for the Ubiquitous Web

WEBlab Group

- **Chair**
  - Michele Colajanni
- **Researchers**
  - Mauro Andreolini
  - Riccardo Lancellotti
- **Post-doc**
  - Alessandro Bulgarelli
  - Claudia Canali
  - Francesca Mazzoni
  - Luca Messori
- **PhD students and grants**
  - Sara Casolari
  - Mirco Marchetti
  - Daniele Gozzi

http://weblab.ing.unimo.it

University of Modena and Reggio Emilia



Goal of the tutorial *(interaction is recommanded)*

- **We have several Web content and services**
- **We have several *Quality of Web-based Services* (QoWS) properties:**
  - Performance
  - Scalability
  - Availability
  - Security
  - Privacy

Which architecture can support Web-based services by satisfying QoWS properties?

Market demands infrastructures to deliver high performance for business-critical applications

- “28% of shoppers who have suffered failed performance attempts said they stopped shopping at the web site where they had problems, and 6% said they stopped buying at that particular company’s off-line store.”

  (Boston Consulting Group, quoted in Infoworld/Computerworld)
- “It takes only 8 ½ seconds for half of the subjects to give up”

  (Peter Bickford, in Netscape/View Source Magazine)
- “Perhaps as much as $4.35 billion in e-commerce sales in the U.S. may be lost each year due to unacceptable download speeds and resulting user bailout behaviors.” (Zona Research)
- “58% of online customers surveyed indicated quick download time as a key factor in determining whether they would return to a web site.”

  (Forrester Research)
- “One of the top three reasons cited by online shoppers for dissatisfaction with a Web site is slow site performance.”

  (Jupiter Communications / NFO Worldwide)
- “At one site, the abandonment rate fell from 30% to 6-8% because of a one second improvement in load time.” (Zona Research)

Zero tolerance for unresolved complaints

- Lost prospective customer
  - If the site did not work, or took too long, your customer may not return for a long time – if ever
- Lost sale
  - If your competitor's site was up and responsive, you may have lost a sale
- Lost customer
  - If this happens repeatedly, you have lost a customer,
  - AND the customer may stop going to associated Web sites and physical locations!
- Lost reputation
  - People talk about poor performance; word spreads …

Goal

- **People are looking for a very few good sites that they can trust!**

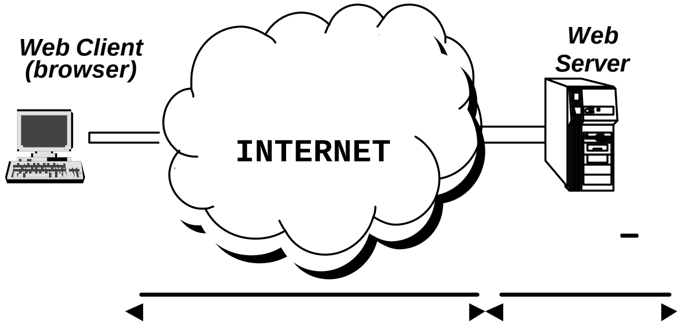
  Even from the performance-availability-security perspectives

➔ "Be sure that your Web site is in that narrow subset!"

Quality of Web-based Services (QoWS)

- **Availability**
  - *Service Level Agreement*: **Web system must be available for X% of times,** e.g.,
    - X = 99% → **7.2** hours/month downtime
    - X = 99.9% → **43** minutes/month downtime
    - X = 99.999% → **26** seconds/month downtime (5 minutes/year downtime)
- **Performance**
  - *Service Level Agreement*: **X% of (all or subset of) Web requests must have a response time less than Y seconds**. Typical measures are 90- or 95-percentile, e.g.,
    - 95% of the requests must have a response time less than 5 seconds

Never forget



Response time = Network time + Server time

Consequence:
“A good researcher in this field has to know networks AND servers”

Instant Web tutorial

WORLD WIDE WEB

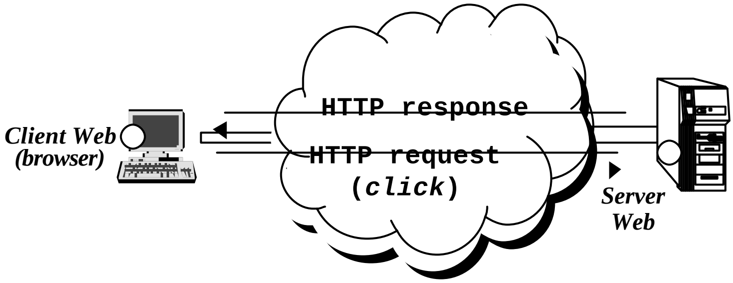
is the

Internet “killer application”

Why?

Request for a Web resource: *user’s view*

User’s actions: *point-click-get*



“The user life is simple, hence the complexity has to be managed by the network/system infrastructure”

Web ingredients behind the scenes

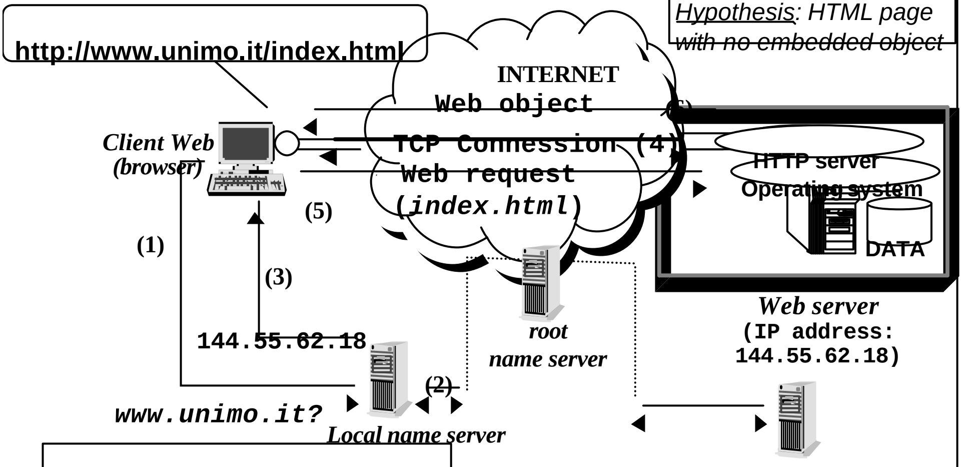
- **Client-server application**

- **Communication and naming (Internet standards)**
  - TCP/IP stack protocol → Internet ? Web
  - Domain Name System (DNS)

- **“Just” three new standards**
  - **URL (resource naming)**
  - **HTML (markup language)**
  - **HTTP (request/reply protocol)**

Request for a Web resource:
more realistic view (not quite real!)

Hypothesis: *HTML page with no embedded object*



http://www.unimo.it/index.html

INTERNET

Web object

(6)

TCP Connession (4)

Web request (*index.html*)

Client Web (browser)

HTTP server

Operating system

DATA

(5)

(1)

(3)

144.55.62.18

(2)

www.unimo.it?

Local name server

root name server

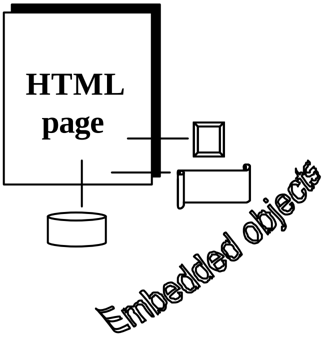
Web server (IP address: 144.55.62.18)

Authoritative name server for www.unimo.it

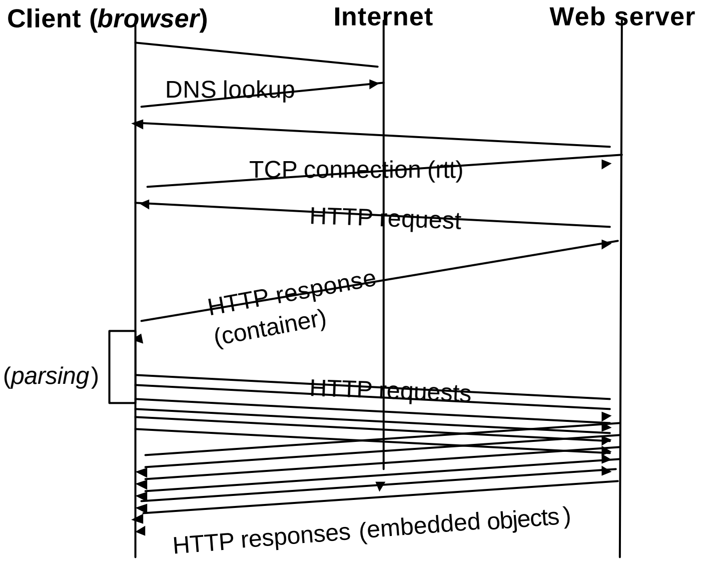
Lookup: (1) - (2) - (3)
TCP Connession: (4)
Request-response: (5) - (6)

Web resource

- **The Web has a simple goal ➔ To download a Web resource from a Web server**
- **Typical Web resource:**
  - HTML page (*container*) + Embedded objects

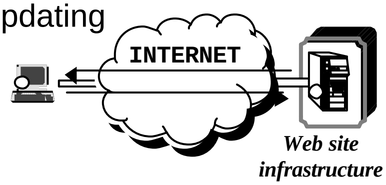


Request for a Web resource: *real view*



The 3 Web generations (*yesterday*)

- **1st generation = *Web publishing***
  - An inexpensive channel for not critical information
  - 90-95% of information represented by text and some images [Arl97]
  - **The large majority of Web sites have static information and few of them adopts dynamic technologies (e.g., CGI)**
  - Occasional maintenance and updating
  - Highly variable performance
  - No guarantee on availability
  - Security not important



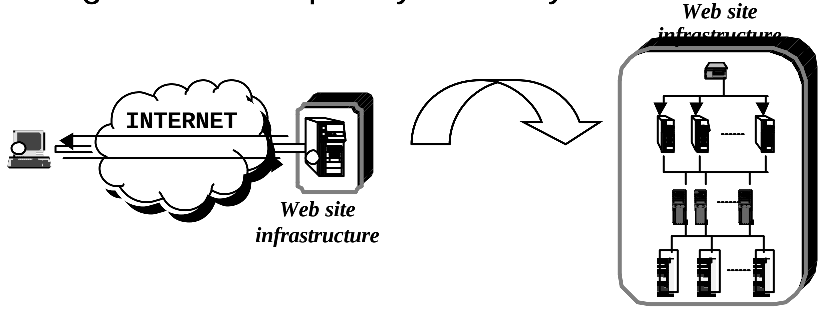
The 3 Web generations (*today*)

- **2nd generation = *Web-based Information Systems***
  - An important channel for critical information
  - Large percentage of dynamically generated contents
  - Direct or *indirect* (say, publicity) costs
  - The quality of services provided by a Web site changes user's view on any company or organization
  - The Web technologies are the preferred interface for many other services ("*My research is different I don't care*" is not acceptable):
    - ♦ Mail, file transfer, blog, resourse management, GRID, …
    - ♦ Access to databases, any source of information, search, …
  - Personalized services, payment services, …

➔

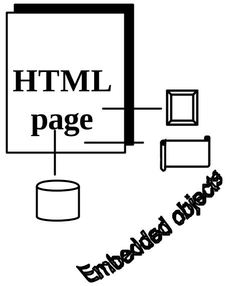
The 3 Web generations (*today*)

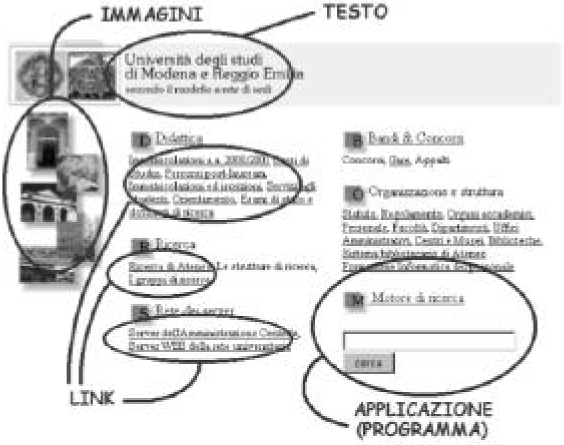
- **2nd generation** = ***Web-based Information Systems***
  - ➔ Necessity of Quality of Web-based services
    - ♦ **Guaranteed performance**
    - ♦ **Availability**
    - ♦ **Security**
  - ➔ Augmented complexity of the system infrastructure



A modern Web resource

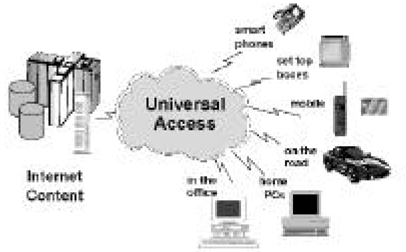
- **Style sheet + any type of *embedded objects (text, image, video, banner, application, form, …)***





The 3 Web generations (*tomorrow*)

- **3rd generation = *Ubiquitous Web***
  - All Web-based services of the second generation:
    - ♦ ***anytime*** (24/7)
    - ♦ ***anywhere***
    - ♦ ***anyway*** (through any device)



First phase (*now*)

- ♦ "Variable" performance and availability requirements
- ♦ Light security

Second phase (*tomorrow*)

- ♦ Guaranteed performance and availability
- ♦ Strong security

Outline of the tutorial

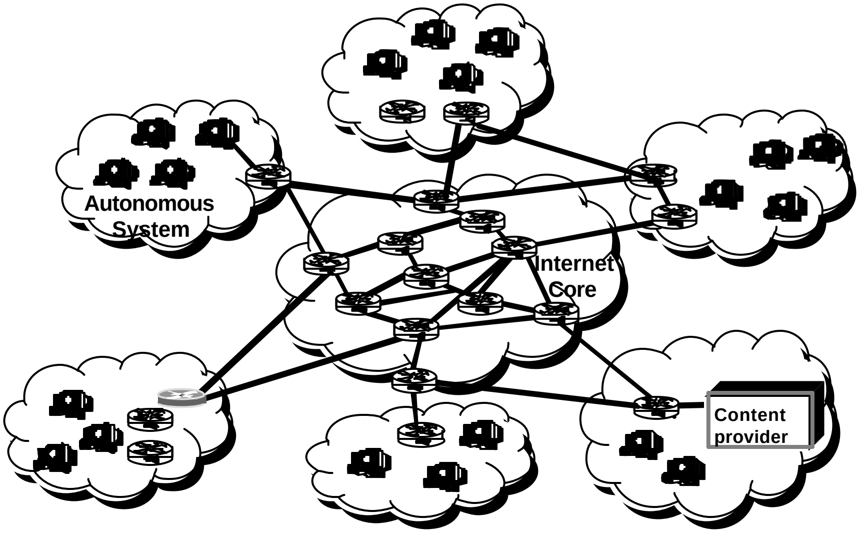
- **Lessons: February 6-7**
  - **2nd Web generation :**

    Web-based information systems

- **Lesson: February 9**
  - **3rd Web generation :**

    Ubiquitous Web-based services

The overall picture



The three important players: *Clients – Network – System*

Recall the main goal

- **We have several <u>Web contents and services</u>**
- **We have several *Quality of Web-based Services* (QoWS) properties:**
  - Performance
  - Scalability
  - Availability
  - Security
  - Privacy

Which **<u>architecture</u>** can support Web-based services by satisfying QoWS properties?

CLIENTS

(→ *WORKLOAD MODELS*)

Importance of workload characterization

- **Aims at reproducing as accurately as possible the characteristics of real traffic patterns**
  - At least the most relevant characteristics of the real workload
- **Not a trivial task as Web traffic exhibits:**
  - **Some peculiar statistical features** (i.e., burstiness and self-similarity)
  - **Less homogeneity than other applications**: large variety, that tends to augment with the increasing number of dynamic requests and different categories of Web sites

Web workload is different from any previous load

- **The Web is a highly variable system**
  - – Geographical location
  - – Day of the week and time of the day
  - – Responses vary across multiple orders of magnitude
- **Workload is heavy-tailed distributed**
  - – Very large values are possible with non-negligible probability
- **Unpredictable nature of information retrieval and service request**
  - – Highly variable load and different access patterns
  - – Difficulty of sizing system capacity to support load spikes
- **Web traffic is bursty in several time-interval scales**
  - – Peak rates are much larger than the average rate

Premise 1: Voice model *vs*. Data model

- **Traditional voice traffic (phone)**
  - – Poissonian arrivals of calls to the telephone system
    - ♦ Arrivals are independent
    - ♦ Inter-arrival times are exponentially distributed
    - ♦ Calls duration are exponentially distributed
- "Voice traffic modeling has proven nothing short of disastrous when applied to data networks, for the simple but profound reason that the rules all change when it is computers and not humans doing the talking"

  [Willinger-Paxson, American Math. Soc., Sept. 1998]
- **The advent of the Web changed everything: decades of analytical results are simply inapplicable now**

Premise 2: Heavy-tailed distributions

- **A random variable X where F(x)=P[X£x], is heavy-tailed distributed (with tail index a) if:**

$$\mathbf{1\text{-}F(x)=Prob[X>x]} \sim \mathbf{c\ x^{-a}}\ \ \mathbf{,}\quad 0< \alpha <2$$

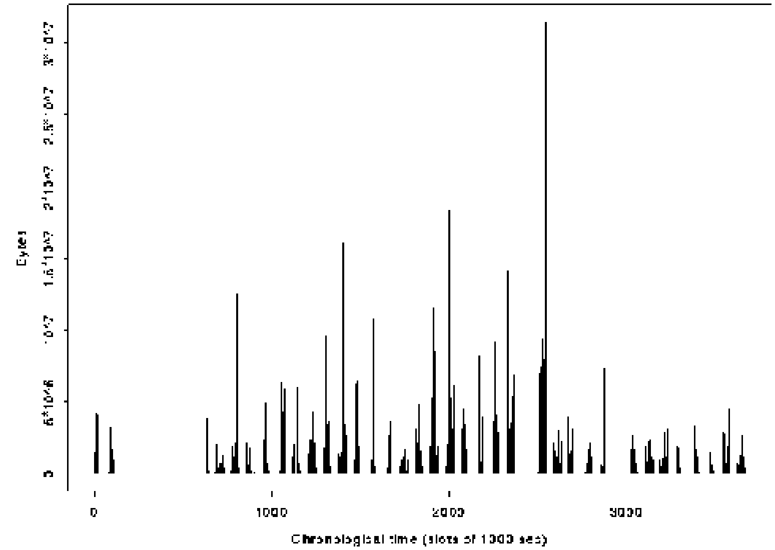
- **Pareto is a simple example of heavy-tailed distribution:**

$$\mathbf{F(x) = 1 – (k/x)^{a}}\ \ \mathbf{,}\quad \alpha,\ k>0\ ,\ \ x\geq k$$

$$\mathbf{f(x) =}\ a\ \ \mathbf{k^{a}\ x^{-a-1}}$$

- Heavy-tailed distributions have infinite variance and, if α<1, even infinite mean
- **Heavy-tailed distributions are characterized by very large variability → the mean is inadequate for representing the system behavior**

Tipical Web traffic (*bytes*)



Issues in service characterization

- **Server workload characteristics depend on the class of the Web site**
- **The category of the Web site affects the distribution of the load over time**
  - E.g., online shopping and B2B sites have different peak and busy load periods

- **<u>The workload for "THE Web site" does not exist</u>**

  For example, IBM identifies five categories of high-volume Web sites

Web site classification

- **"Old plain" (simple browsing) Web sites**
  - Content is mainly static

Large volumes of dynamic transactions for remaining classes

- **Publishing Web sites**
  - Content change frequently
    - But fairly static information sources
  - Security considerations are minor
- **Online shopping sites**
  - Content can be relatively static and dynamic
  - Significant amount of secure transactions
  - Typical requests such as browse, search, select, add, and pay
- **Customer self-service sites (e.g., e-banking, e-ticketing)**
  - Complexity of transactions
  - Multiple data sources, consistency issues are significant
  - Significant amount of secure transactions

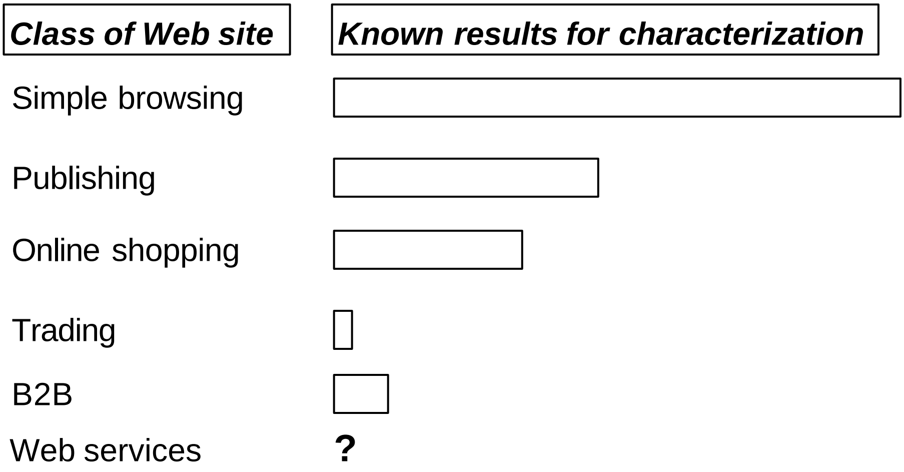
Web site classification *(cont.)*

- **Trading sites (e.g., stock exchange market, e-bay)**
  - A great deal of rapidly changing content
  - Complexity of transactions
    - Most transactions interact with back-end servers
  - Significant amount of secure transactions
  - Typical requests such as browse, select, bid
- **Business-to-business / Web services sites**
  - Complexity of transactions (substantial purchasing activity)
  - Multiple data sources, consistency issues are significant
  - Nearly all transactions are secure

Dynamic content characterization

- **Studies on Web workload characterization have been focused on static content (not e-commerce Web traffic)**
- **Still few studies consider Web sites with prevalent dynamic (and personalized) Web content**
  - Difficulties in obtaining data for dynamic content characterization
    - Unavailability of representative data
    - Privacy, competitive concerns
  - No consensus as to what constitutes a representative dynamic workload (“The mythical Web site”)
- **Even some conclusions on static content need to be revisited** (e.g., *academic sites* vs. *commercial sites*)

Classification of Web sites and workload characterization



Characterization of simple browsing sites

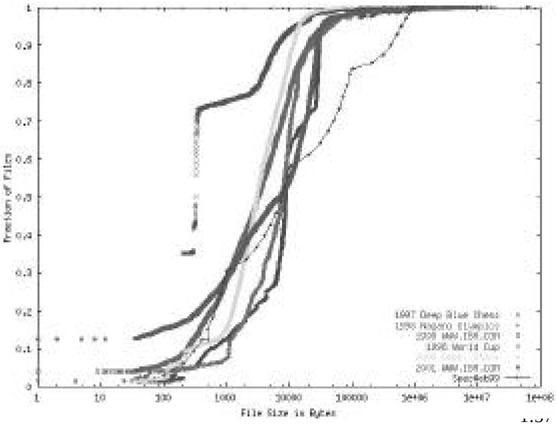
- **Many research results capture the characteristics of static Web content (including high-volume Web sites)**

  Arlitt and Williamson, *IEEE/ACM Trans. on Networking*, Oct. 1997
  Crovella and Bestavros, *IEEE/ACM Trans. on Networking*, Dec. 1997
  Barford and Crovella, *Proc. Performance/ACM Sigmetrics 1998*, July 1998
  Iyengar et al., *World Wide Web Journal*, Mar. 1999
  Pirolli and Pitkow, *World Wide Web Journal,* Mar. 1999
  Barford and Crovella, *Proc. ACM Sigmetrics 1999*, May 1999
  Liu et al., *Performance Evaluation*, 2001

- **A wide range of investigated characteristics**
  - *User behavior characteristics*
    - Session and request arrivals, clicks per session, request interarrival times
  - *Object characteristics*
    - Sizes, content types, resource popularity, resource changes, temporal locality, embedded objects

Simple browsing sites: object characteristics

- **Object size**
  - Sizes have Lognormal distribution for body, Pareto for tail (some controversy)
  - Most responses are small, most of the bytes are from large transfers
- **Object popularity**
  - Zipf-like distribution



[Source : E. Nahum , *Proc. WCW 2002*]

Characterization of publishing sites

- **Some studies**

Arlitt and Jin, *IEEE Network*, May 2000
Padmanabhan and Qiu, *Proc. ACM Sigcomm*, Aug. 2000
Shi et al., *Proc. WCW 2002*, Aug. 2002
Shi et al., *Proc. IEEE Globecom 2002*, Nov. 2002

- **Object characteristics**
  - Exponential distribution of objects sizes embedded in a dynamic page (not heavy-tailed)
    - ♦ Some controversy about large transfers
- **Peculiar characteristics of dynamic content**
  - Freshness time (Weibull distribution)
  - Content reusability

Characterization of online shopping sites

- **Some studies**

Menascé et al., *Proc. ACM Conf. on Electronic Commerce,* Oct. 2000
Arlitt ed al., *ACM Trans. on Internet Technology*, Aug. 2002
Vallamsetty et al., *Proc. Wecwis 2002*, June 2002
Shi et al., *Proc. IEEE Globecom 2002*, Nov. 2002

- **E-commerce traffic is significantly more complex than simply-browsing traffic**
    - ♦ A variety of activities
    - ♦ A high level of Online Transaction Processing activity
    - ♦ A high proportion of dynamic requests
- **Arrival characteristics**
    - – Arrival traffic is more bursty than normal
- **Object characteristics**
    - – ***Size*** of transferred objects is ***not heavy-tailed***
        - ♦ But response times show heavy-tailed behavior (due to server processing and back-end data retrieval times
    - – Popularity of search terms (*Zipf-like* distribution)
    - – Freshness time (*Bimodal* distribution)

Characterization of trading and B2B sites

- **Very few preliminary results**
    - – For trading sites

Menascé et al., *Proc. ACM Conf. on Electronic Commerce,* Oct. 2000

- – For B2B sites

Vallamsetty, Kant, et al., *Proc. Wecwis 2002*, June 2002
Vallamsetty, Kant, Mohapatra, *Electronic Commerce Research,* Jan. 2003

- B2B sites
    - – Heavy-tailed distribution of response times
    - – Lower number of embedded objects
    - – Secure pages are simpler
- Lack of back-end transactional characteristics and their relationship with front-end transactions

References for benchmarking models

- TPC-W (www.tpc.org)
- SPEC-Web (www.spec.org)

SYSTEM

(*SOFTWARE TECHNOLOGIES*)

(*Logical*) levels of a Web-based service

- User interface
- Presentation logic
- Application logic (or *business logic*)
- Data logic

Important note

We have:

- Logical levels
- Processes
- Computers

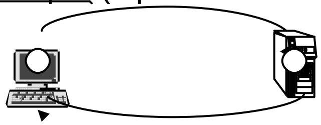
You should not confuse the “logical levels”
with the “processes” that implement the
functions of the logical levels and
with the “computers” that run the processes

Alternatives for mapping 4 logical levels on processes

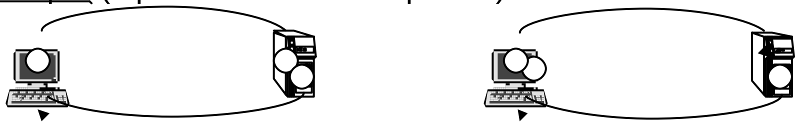
- ***All on one process (theorical)***
- **On two processes**
  - **Client process**: implements the "user interface"
  - **Server process**: implements 3 levels (*presentation*, *application*, *data*)
- **On three processes**
  - **Client process**: implements the "user interface"
  - **1st Server process**: implements *presentation* and *application* levels
  - **2nd Server process**: implements *data* level
- **On four processes**
  - **Client process**: implements the "user interface"
  - **1st Server process**: implements *presentation* level
  - **2nd Server process**: implements *application* level
  - **3rd Server process**: implements *data* level

Alternatives for mapping 2 or more processes on computers

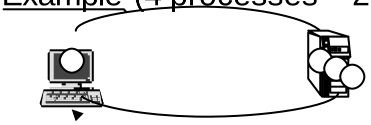
Example (2 processes – 2 computers)



Example (3 processes – 2 computers)



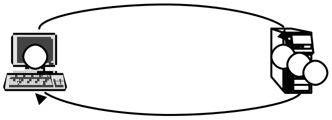
Example (4 processes – 2 computers)



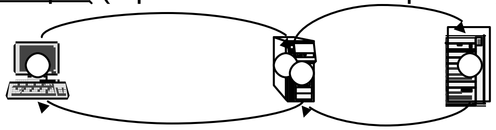
➔

Alternatives for mapping 4 processes on 2 or more computers

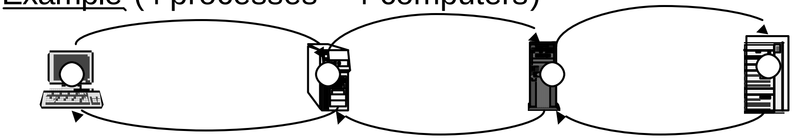
Example (4 processes – 2 computers)



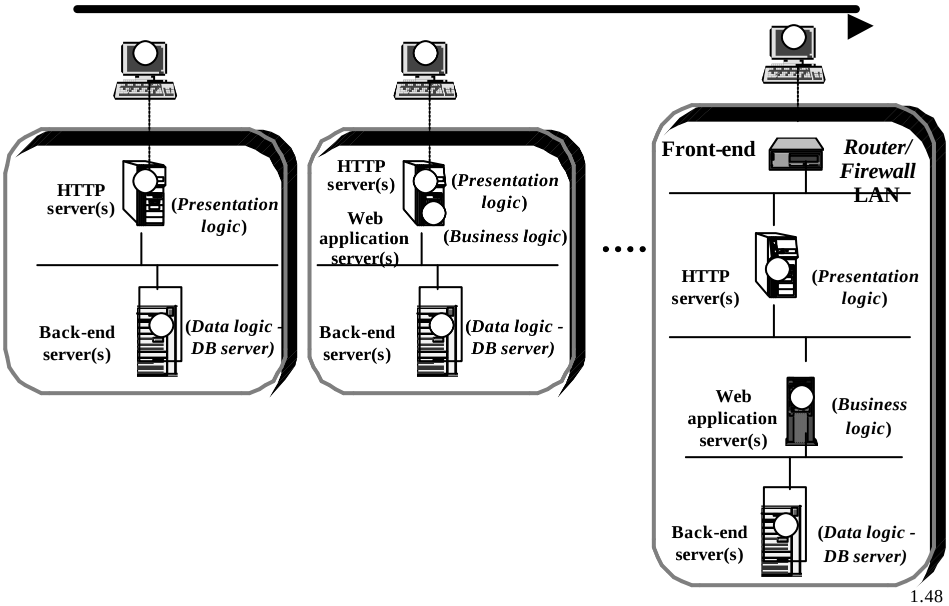
Example (4 processes – 3 computers)



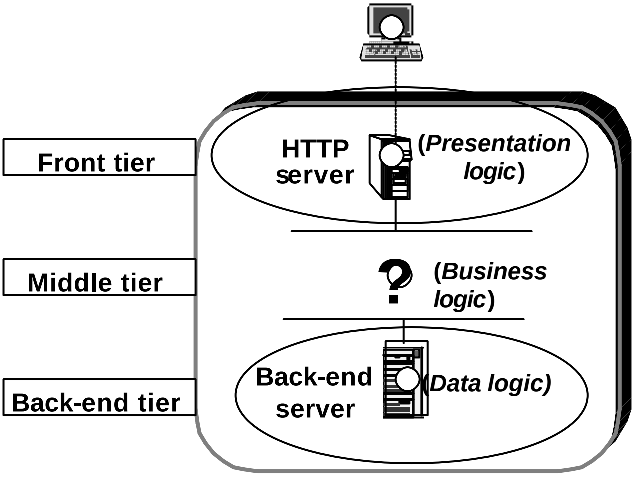
Example (4 processes – 4 computers)



Evolution of software architettures



The first levels: *Presentation & Data*



Front tier

Middle tier

Back-end tier

HTTP server

(*Presentation logic*)

(*Business logic*)

Back-end server

(*Data logic)*

Presentation logic

- **Today, it is an interface between the user interface and the business logic**
- **Implementations:**
  - **HTTP server**
    - ♦ Apache
    - ♦ Microsoft IIS
  - **(possible) Business-logic plugins**

Data logic

- **It manages information on disks**
- **Implementations:**
  - **Database Management systems**
    - ♦ Microsoft Sql server
    - ♦ IBM DB2
    - ♦ Oracle
    - ♦ MySql (open source)
    - ♦ Postgress (open source)
    - ♦ …

Business logic

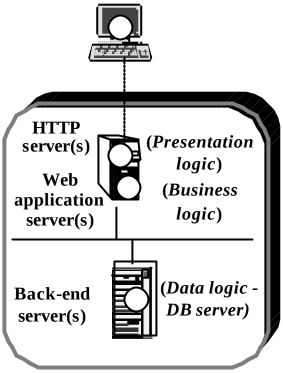
- **CONTINUOS EVOLUTION OF THE MIDDLE-TIER**
  - Technologies external to the HTTP server process
  - Scripting languages
  - Distributed object technologies

Middle tier technologies (I)

- **External technologies**
  - **Common Gateway Interface** (**CGI**): **C** or **Perl** scripting languages
  - HTTP server activates a new process for the CGI program (→ program copy, memory allocation, different environment for variables, etc.)

  ➔ **NOT SCALABILE SOLUTION**

  (inappropriate for a server that must manage many client requests)



CGI-based
(Separation between *Presentation* and *Business* logic)

First evolution step

- **If the scalability is the main problem**
- **If the limit to the scalability is due to the necessity of activating a new process for each dynamic request**

⇩

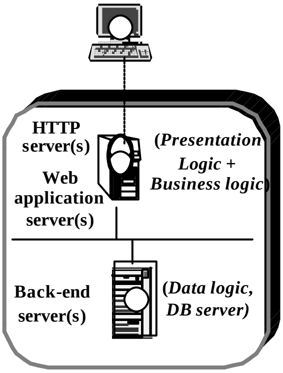
- **Avoid the creation/activation of a new process → Scripting technologies**

Middle tier technologies (IIa)

- **Server API** (Netscape NSAPI, MS ISAPI): Shared libraries that can run in the HTTP server space

  → may serve multiple requests with no novel process

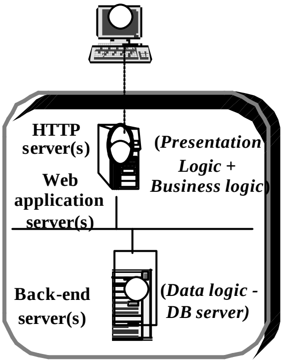
  → limite portability (strictly related to the HTTP server) and *thread unsafe*
- **ColdFusion**
- **Mod_Perl**: Apache module that can interpetrate Perl scripts within the HTPP process


HTTP server(s)
Web application server(s)
(*Presentation Logic + Business logic*)
Back-end server(s)
(*Data logic, DB server)*

(Integration of *Presentation* and *Business* logic)

Middle tier technologies (IIb)

- **Scripting languages (code that is inserted into the HTML page and that is interpreted by the HTTP server)**
  - **Active Server Pages** (**ASP**, Microsoft): VBScript → Solution just for IIS server
  - **Java servlets** (Sun): server side programs that can serve multiple HTTP requests through one process (it is like a multi-thread applet executed on the server side) → based on JVM: portable, but not efficient
  - **Java Server Pages** (**JSP**, Sun): it may integrate HTML, Java and JavaBean
  - **PHP** → "Programming Language of the year 2004"


HTTP server(s)
Web application server(s)
(*Presentation Logic + Business logic*)
Back-end server(s)
(*Data logic - DB server)*

(Integration of *Presentation* and *Business* logic)

Second evolution step

- With the continuos improvement of hardware platforms (CPU, disk, RAM, cache), performance remains an important issue of Web-based applications, but we want also integrate more complex applications and information systems through the Web technologies

- The middle-tier complexity augments (not just an access to one DB, but accesses to multiple DBs, integration with XML files, directory services, etc.) → necessity of a real and complex middle-tier software for ***business logic***

- **With more complex applications → portability, flexibility, maintenance, modularity, software reuse, etc. become more and more important**

➔

Second evolution step

- **Scripting technologies that aim mainly to improve performance at the price of a more embedded software, do not satisfy software requirements of the most modern Web-based applications**

- **A complex *business logic* cannot be integrated into the *HTTP server* process**

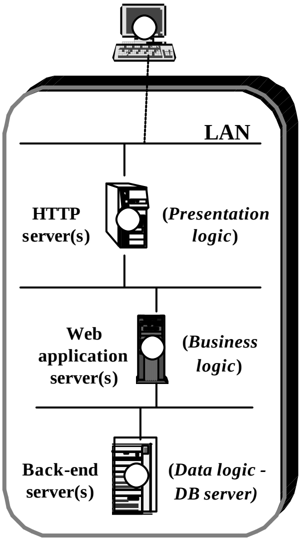
  NOTE: It's not a return to the CGI technologies. The goal is quite different!

- **Which is the technology that better satisfies the principles of:**
  – Modularity
  – Portability
  – Maintenance and software reuse

?

Middle tier technologies (III)

- **Distributed objects technologies**
  - **Each object has an interface (together with its public methods)**
  - **The code may run where it is necessary (max portability)**
  - **Many applications can utilize common *business objects***
  - **Better maintenance properties**



DISTRIBUTED OBJECTS
(Separation between *Presentation* and *Business* logic)

Distributed objects technologies

- **CORBA (standard *de jure* of the ’90, *now in bad shape*)**

- **Open source**

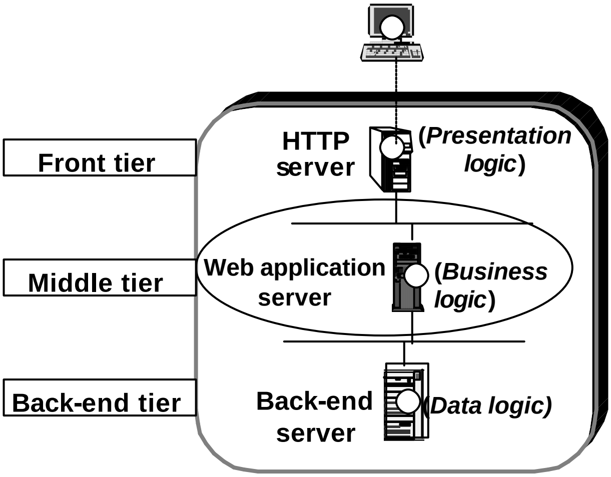
  Java2 Enterprise Edition (J2EE) - SUN

- **Proprietary**

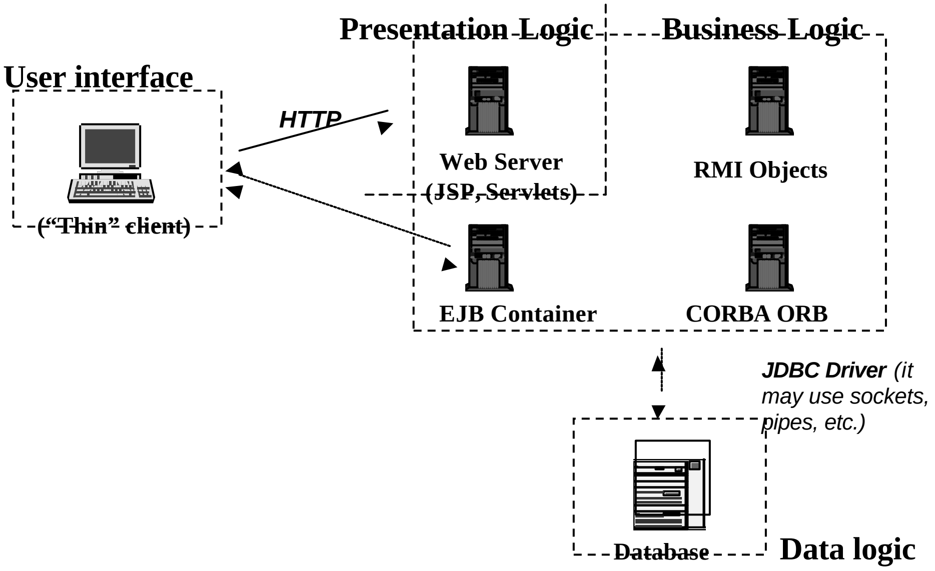
  .NET - Microsoft

The *Web Application Server* was born

Web Application Server: **“a Web server with a business logic”**



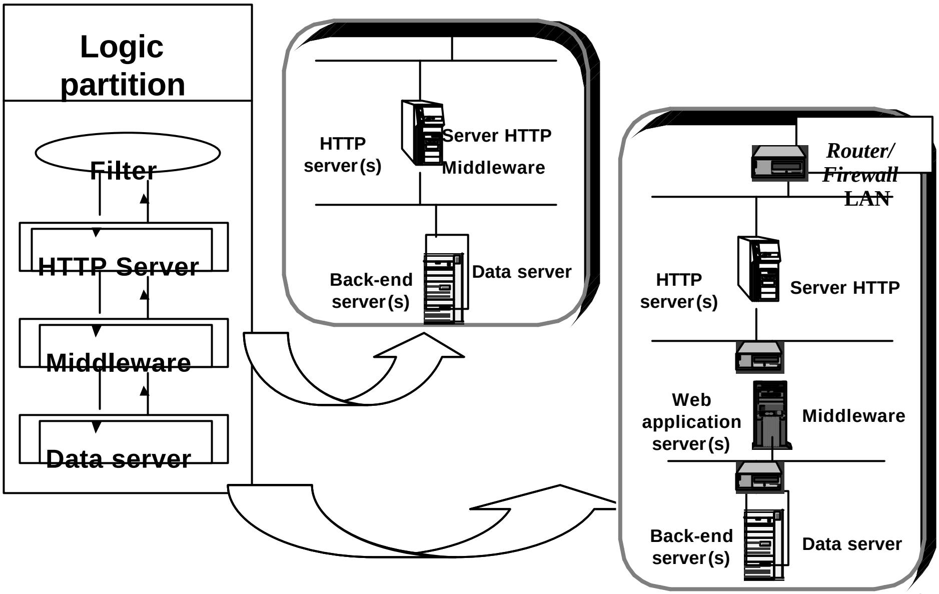
Example of *n-tier* J2EE architecture



Commercial WAS

- IBM Web Sphere
- BEA Web logic
- Inprise AS
- …

Multi-tiers: not only for interoperability



Logic partition

Filter

HTTP Server

Middleware

Data server

HTTP server(s)

Server HTTP

Middleware

Back-end server(s)

Data server

Router/ Firewall

LAN

HTTP server(s)

Server HTTP

Web application server(s)

Middleware

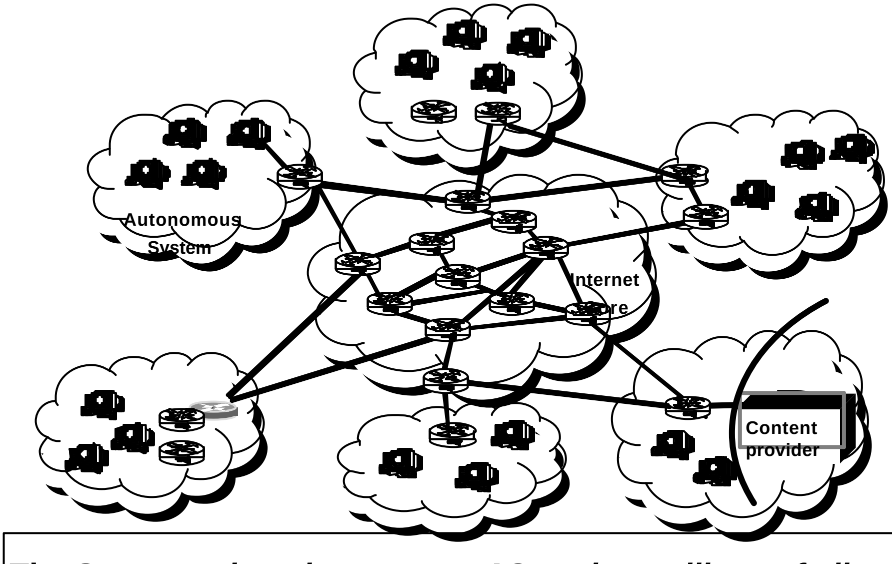
Back-end server(s)

Data server

SYSTEM

(*INFRASTRUCTURES*)

The overall picture



The System: ***alone in a remote AS against milions of clients***

“Hot” Web sites

Yahoo, Microsoft, Google, CNN, … **(*>50 Milions hits/day*)**

Event	Hits period	Peak hits/day	Peak hits/minute
NCSA server (Oct. 1995)		**2 Milions**	
Olympic Games 1996 (Atlanta, 1996)	**180 Milions**	**8 Milions**	
NASA Pathfinder (July 1997)	**942 Milions (14 days)**	**40 Milions**	
Olympic Winter Games (Japan, 1998)	**634.7 Milions (16 days)**	**55 Milions**	**110.000**
FIFA World Cup (France, 1998)	**1.350 Milions (90 days)**	**73 Milions**	**209.000**
Wimbledon (July, 1999)	**942 Milions (14 days)**	**125 Milions**	**430.000**
Wimbledon (July, 2000)		**282 Milions**	**964.000**
Sydnwy Olympic Games (August 2000)		**875 Milions**	**1.200.000**

Good performance is a challenge

- **24x7 availability and geographic distribution; expectation of universal access**
- **A shared network resource**
- **No control over customers’ environment**
- **Multiple servers, which may be geographically distributed, participate in a single user interaction**
- **Dynamic, complex content**
- **Poor support for session structures**
- **Potentially massive peak volumes**
- **Difficult to predict workload mix**

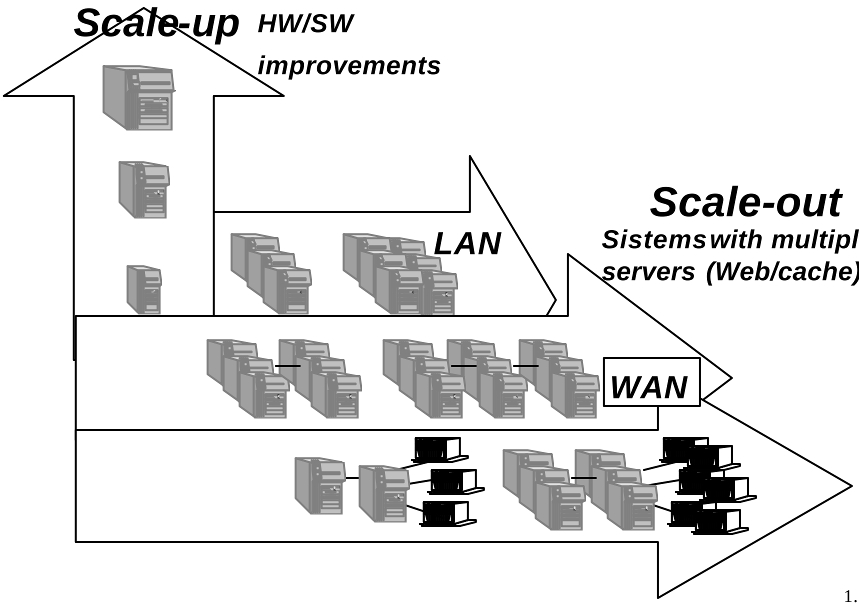
Metrics for the future

- Integrated data, voice and video services
  - 5 nine Reliability 99.999 (<5 min downtime/yr)
  - Server Speed (200 billion ops/sec)
  - Server Connectivity (>1 Gigabyte/sec)
  - Systems Reconfiguration (<1 min delay)
  - Access to petabytes of data (<0.25 sec)
  - Customizing an Application (<1 hr implementation)
  - Strong authentication for access

What a computer engineer does when he/she needs power?

- **REPLICATION**
- **CACHING**

Pressure on Web-based architectures



Scale-up *HW/SW improvements*

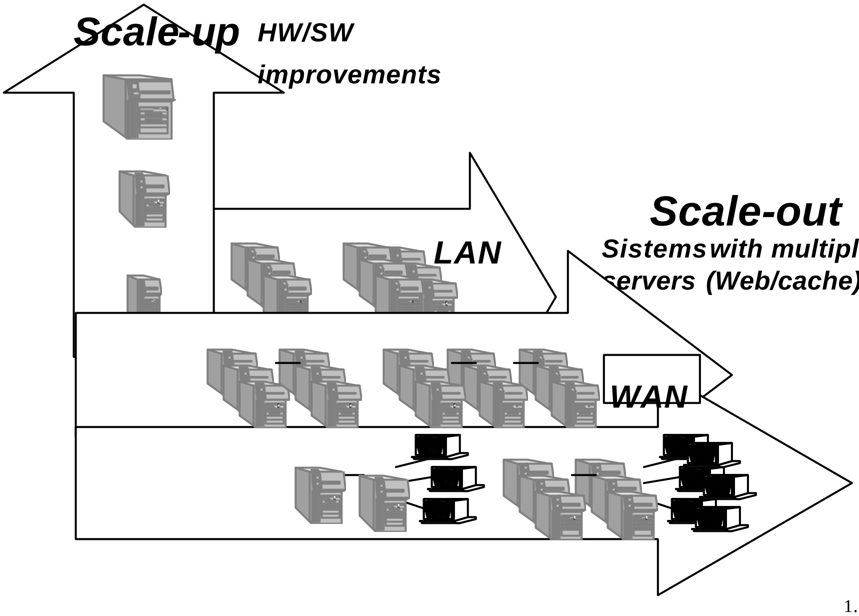
Scale-out

Sistems with multiple servers (Web/cache)

LAN

WAN

Pressure on Web-based architectures



Scale-up *HW/SW improvements*

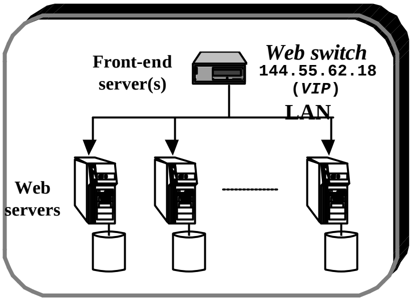
Scale-out

Sistems with multiple servers (Web/cache)

LAN

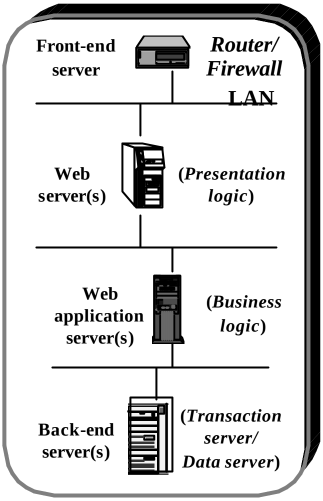
WAN

Web cluster architectures



Horizontal replication

Vertical replication



With all possible combinations …

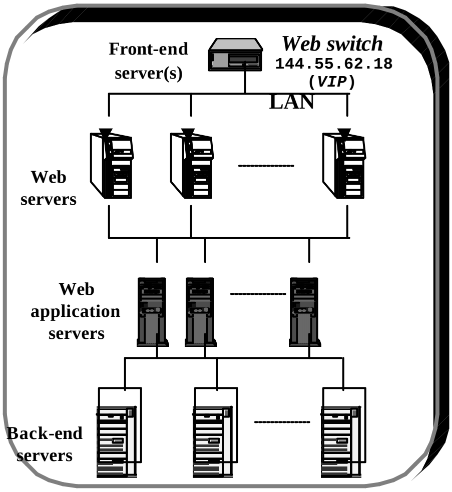
Web cluster architecture:
A myriad of (*complex*) technologies

Network/OS technology

Web server technology

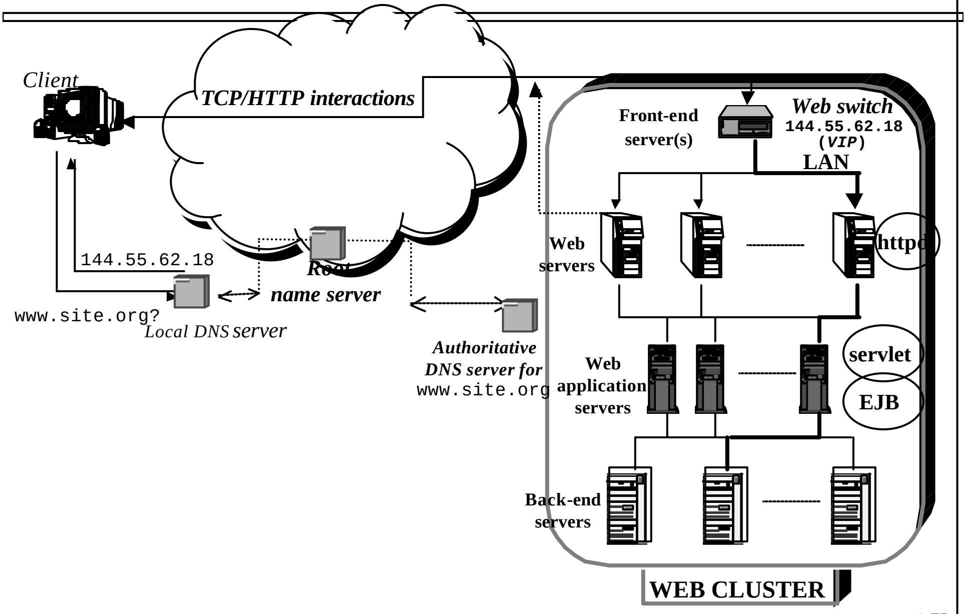
Middleware technology

Database technology

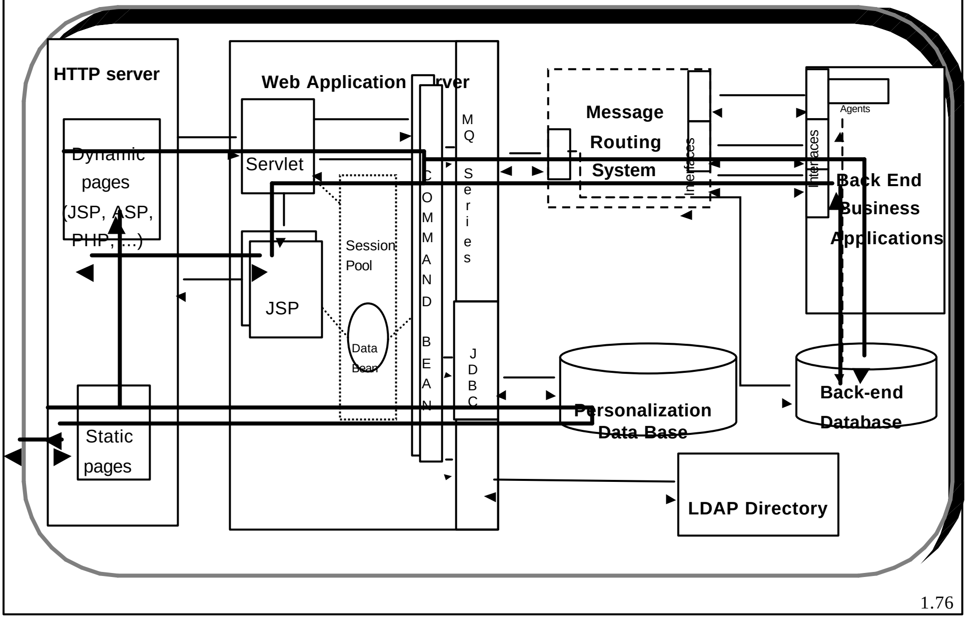


WEB CLUSTER

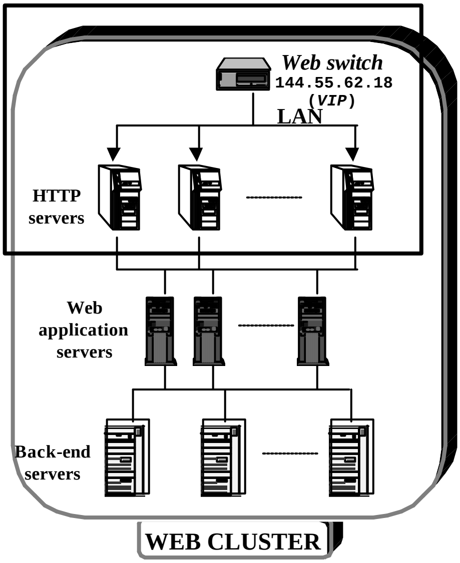
Interaction with a Web cluster architecture



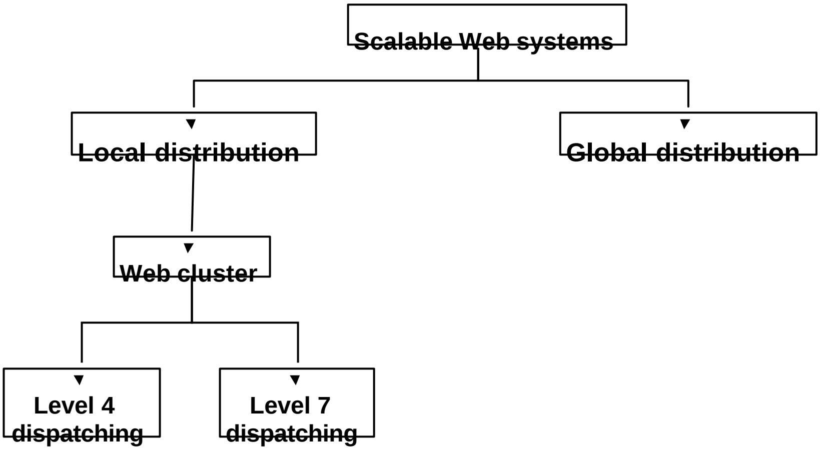
A more complete picture



Focus



Taxonomy



Scalable Web systems

Local distribution

Global distribution

Web cluster

Level 4 dispatching

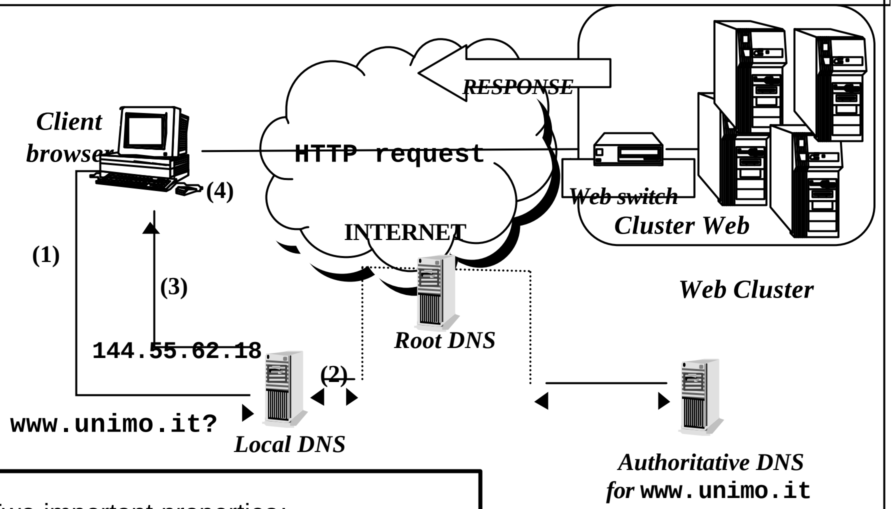
Level 7 dispatching

Web cluster

- ***Web site that is implemented on a parallel or locallaly distributed architecture***
- Web site address
  - One hostname (e.g., "www.unimo.it")
  - One IP address (*virtual IP address*)
- Front-end ***Web switch*** (its IP address is visible and corresponds to the Web site IP address)
- All internal servers have hidden IP addresses

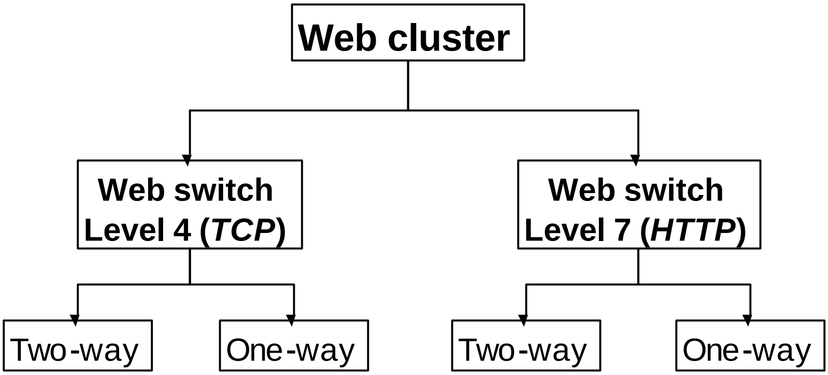
V. Cardellini, E. Casalicchio, M. Colajanni, P.S. Yu,
"The state of the art in locally distributed Web-server systems",
***ACM Computing Surveys*, Vol. 34, No. 2, pp. 263-311, June 2002**

HTTP request to a Web cluster



Two important properties:
- Web switch features
- Response packet flow (*server-to-client*)

Web cluster alternatives



All *inbound* packets reach the Web switch

Two-way: even *outbound* packets pass through the Web switch

One-way: *outbound* packets pass through a different Internet connection

Cluster Web switch

It is a network component with dispatching role

- Mapping VIP into server IP addresses
- Centralized and fine-grain distribution
- Implementations
  - ♦ special-purpose hardware device
  - ♦ software module at kernel level (*special-purpose* operating system)
  - ♦ software module at application level (*general-purpose* operating system)
- Other possible functions:
  - ♦ firewall

"Companies like the centralized control of a Web cluster"