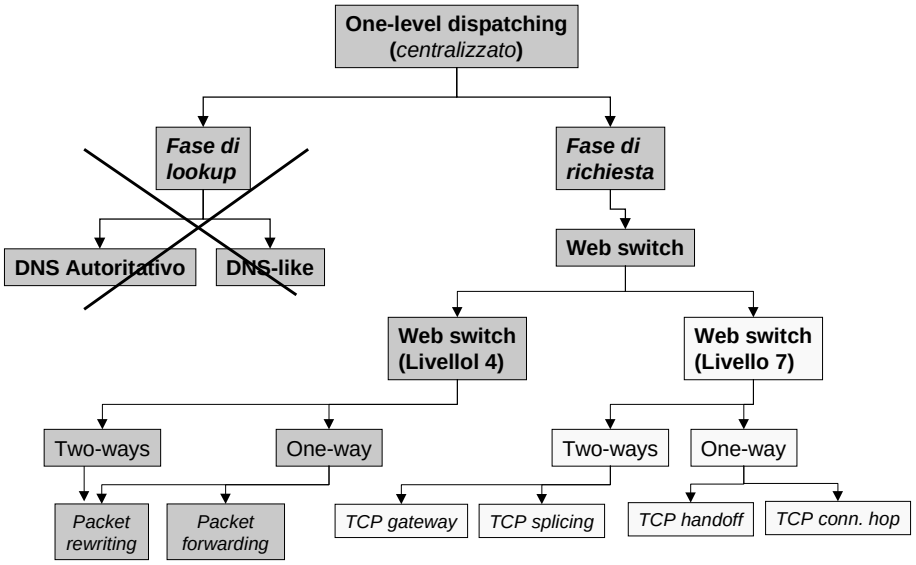
Architetture per Web cluster

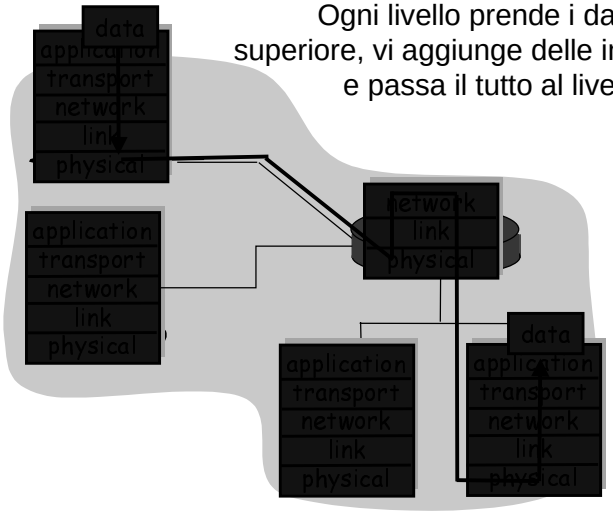


Meccanismi di dispatching

- **Livello 4**
  - Livello connessione TCP
  - Algoritmi di distribuzione *content blind*
  - Algoritmi statici e dinamici (*state dependent*)
- **Livello 7**
  - Livello connessione applicativa (HTTP)
  - Algoritmi di distribuzione *content aware*
  - Algoritmi statici e dinamici

Comunicazione *fisica* tra i livelli

Ogni livello prende i dati dal livello superiore, vi aggiunge delle informazioni e passa il tutto al livello inferiore



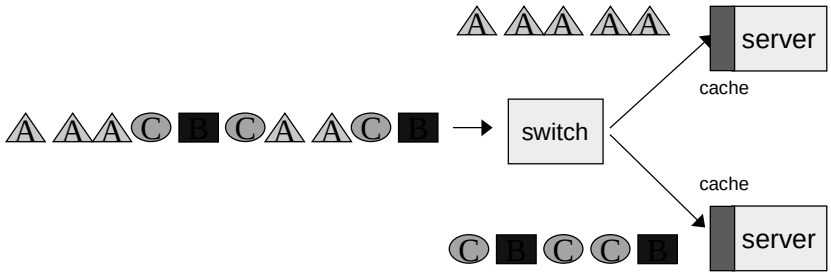
Algoritmi client info aware

- Identificatori di sessione
  - Richieste HTTP con stesso ***SSL id*** o stesso ***cookie*** assegnate allo stesso server
- Content partition (statico)
  - Contenuto partizionato tra i server rispetto al **tipo di file** (HTML, imaggini, contenuto dinamico, audio, video, …)
    - Scopo: utilizzare server specializzati per contenuti differenti
  - Contenuto partizionato tra i server rispetto alla **dimensione di file** (soglie dinamiche) [Har99]
    - Scopo: aumentare load balancing
  - Insieme dei file partizionato tra i server mediante una funzione hash
    - Scopo: aumentare il *cache hit rate* nei server Web

Algoritmo *client-server info aware*

Locality Aware Request Distribution (LARD)*

- considera sia il tipo di richiesta/servizio sia lo stato di carico dei server Web
- ha l'ulteriore obiettivo di aumentare il *cache hit rate* dei server Web



* S. Pai et al., "Locality-aware request distribution ….", 8th ACM ASPLOS, 1998

Proposta: *Client Aware Policy*

Sfrutta informazioni relative alla tipologia della richiesta da assegnare:

- Richiede un meccanismo di classificazione dinamica delle richieste effettuabile in base al tipo di servizio (URL)
  - *CPU bound* (es., crittogtrafia)
  - *Disk bound* (query a database)
  - *Network bound* (download di grandi file)
- Scopo: ripartire le richieste *CPU/disk/network bound* tra tutti i server

Client Aware Policy (CAP o MC-RR)

- **Distinzione fra più classi di servizio**

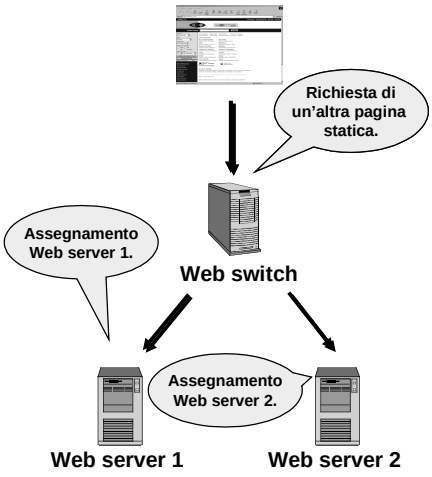
Web Publishing

Web Transaction

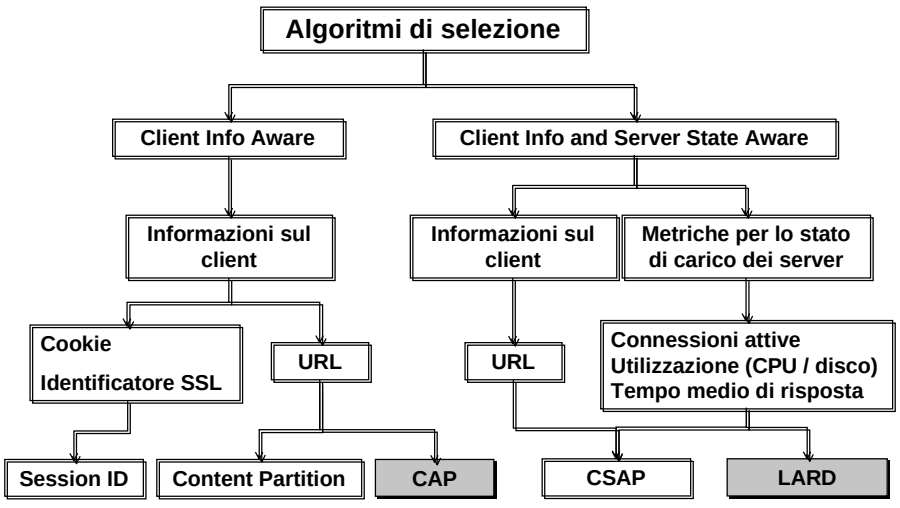
Web Commerce

Web Multimedia

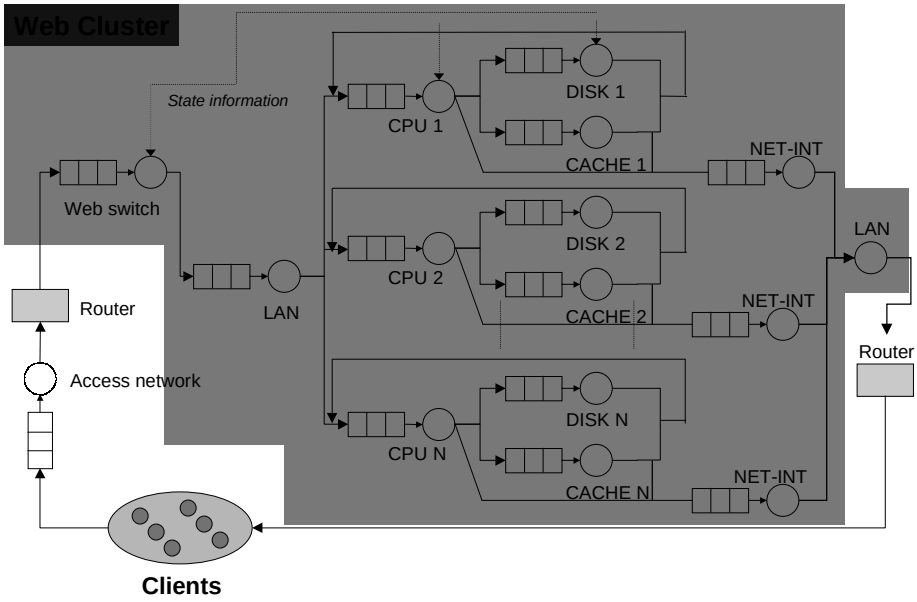
- **Ciascuna classe di servizio viene schedulata in modalità round robin sullo stesso insieme di server**



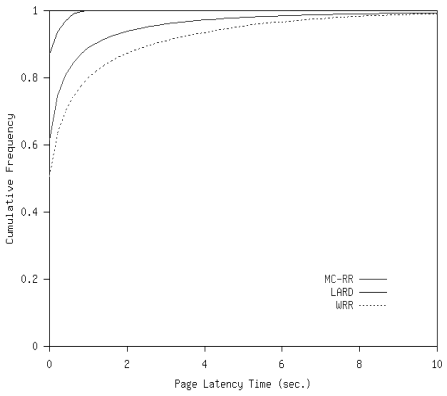
Algoritmi di Livello-7



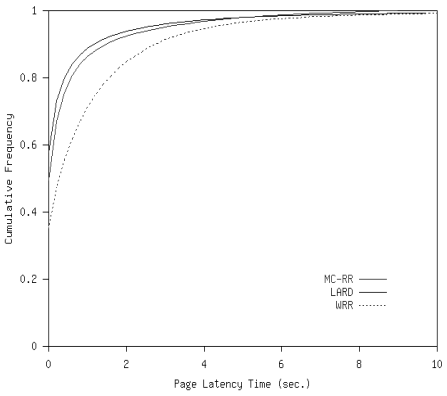
Modello di simulazione



Risultati simulativi: Modello di carico “light”

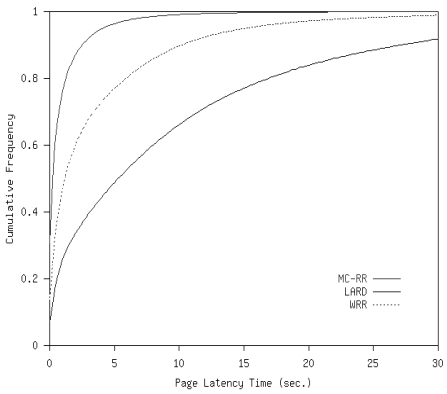


Richieste statiche

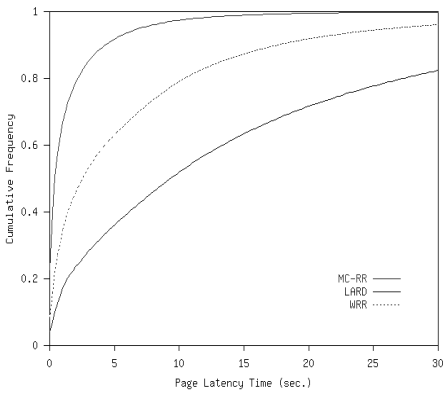


Richieste statiche + dinamiche

Risultati simulativi: Modello di carico “heavy”

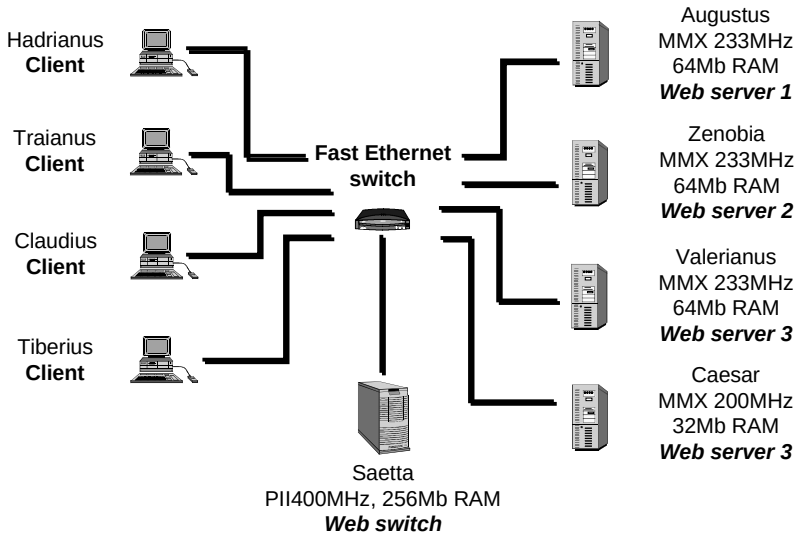


CPU- e disk-bound



CPU-, disk- e CPU-disk-bound

Architettura del prototipo Roma7



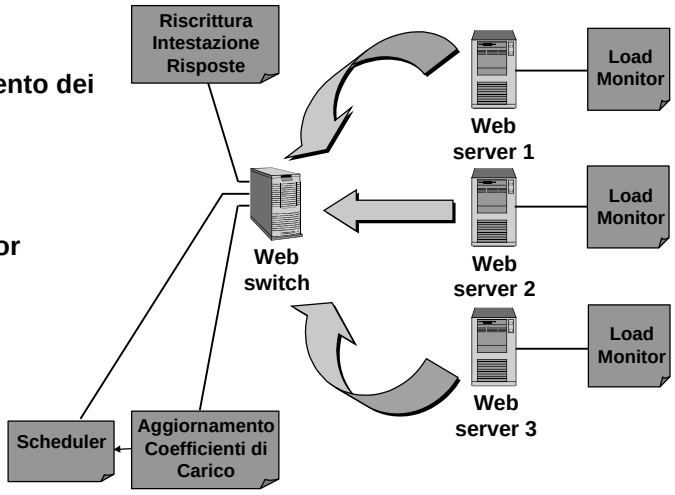
Componenti del sistema

Web switch

- **Scheduler**
- **Aggiornamento dei carichi**

Web server

- **Load Monitor**



Scenari di test

Tool Webstone (modifiche apportate)

- Introduzione dello “user think time”
- Modellazione struttura di una risorsa Web (Pagina HTML + Oggetti incorporati)

Scenari di workload

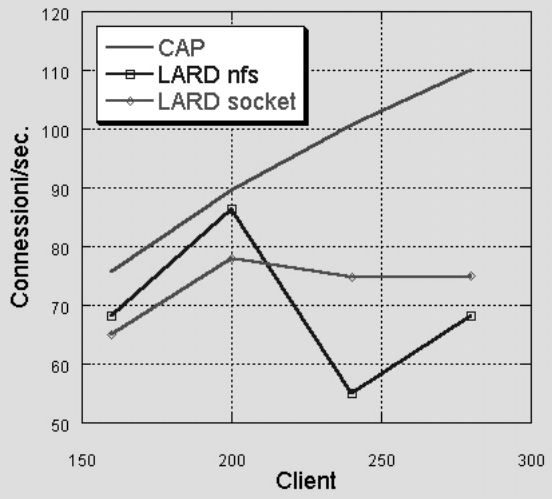
- Light Workload (90% statiche, 10% dinamiche leggere)
- Heavy Workload (90% statiche, 10% dinamiche pesanti)

Metriche utilizzate

- Connessioni / sec. aperte dal cluster
- Mbit / sec. in uscita dal cluster
- Tempo di risposta (sec.) del cluster
- Utilizzazioni medie CPU server e switch

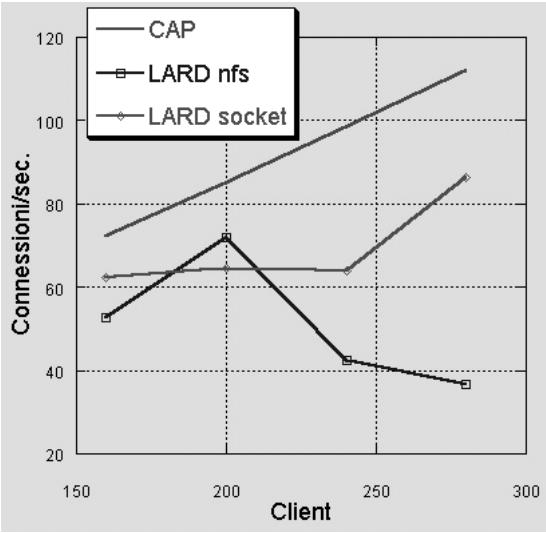
Risultati sperimentali: Modello di carico “light”

- **La politica CAP mostra una spiccata scalabilità**
- **La politica LARD con monitor NFS manda in saturazione il cluster**
- **La politica LARD con monitor socket è migliore rispetto alla LARD con NFS**
- **L’andamento della LARD con NFS è dovuto ad un cattivo bilanciamento del carico**



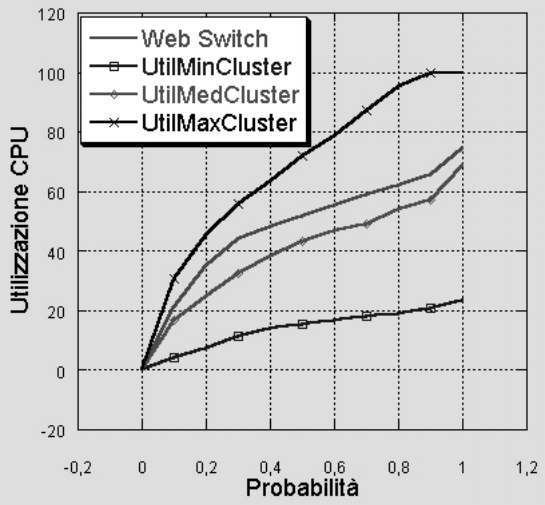
Risultati sperimentali: Modello di carico “heavy”

- **All’aumentare del carico, aumenta il divario fra CAP e LARD**
- **La saturazione anticipata della LARD con NFS è dovuta all’overhead di comunicazione interna e ad un pessimo bilanciamento del carico**



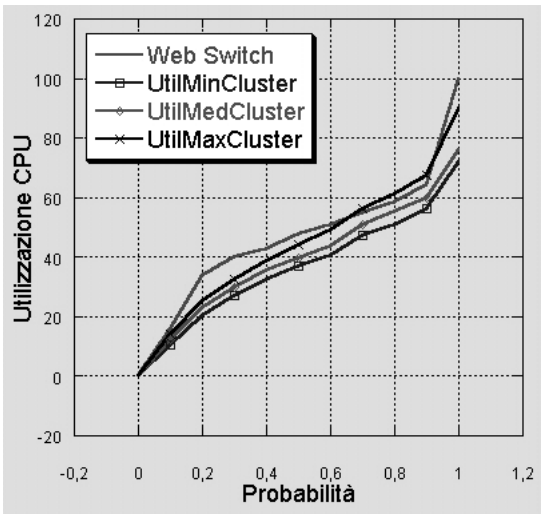
Risultati sperimentali: Bilanciamento del carico

- **Studio delle curve di distribuzione delle utilizzazioni**
- **Esperimento con:**
  - dispatcher LARD NFS
  - 200 client
  - carico leggero
- **Il bilanciamento del carico non è buono**



Risultati sperimentali: Bilanciamento del carico

- **Studio delle curve di distribuzione delle utilizzazioni**
- **Esperimento con:**
  - dispatcher CAP
  - 200 client
  - carico leggero
- **Il bilanciamento del carico è ottimo: le curve di utilizzazione sono molto vicine fra loro**



Prototipi/prodotti Web cluster L7

Two-ways		One-way	
TCP gateway	***TCP splicing***	***TCP handoff***	***TCP conn. hop***
• **IBM** Network Dispatcher CBR [IBMND]	• [Coh99] • **Alteon Web Systems** [Alt] • **ArrowPoint** [Arr] • **Foundry Nets'** ServerIron [Fou]	• **LARD** [Pai98] • [Aro99]	• **Resonate's** Central Dispatcher [ResCD]

Caratteristiche Web cluster

- **Architetture alternative**
  - Level-4 Web switch (*Content information blind*)
  - Level-7 Web switch (*Content information aware*)
- **Principali vantaggi**
  - Controllo a grana fine sull'assegnamento richieste
  - Elevata affidabilità (*availability*, *sicurezza*)
- **Principale svantaggio**
  - Scalabilità limitata dalla banda di accesso ad Internet (es., T3 ≈ 45 Mbps)

Argomenti

0. Motivazioni e background
1. Sistemi Web ad alte prestazioni distribuiti localmente
2. **Sistemi Web a Qualità del Servizio garantita (*cenni*)**
3. Sistemi Web ad alte prestazioni distribuiti geograficamente

Quality of Service (QoS)

- **Network quality**
  - La latenza tra punti specificati della rete non sarà inferiore ad un certo valore
- **Service quality**
  - La rete sarà funzionante il 99% del tempo (7.2 ore/mese inattività)
  - La rete sarà funzionante il 99.9% del tempo (43 minuti/mese inattività)

Esempio (reale) di *Service Level Agreement*

- Round-trip inferiore a **85ms** all'interno dell'Europa e all'interno del Nord America
- Round-trip inferiore a **120ms** per collegamenti transatlantici tra Londra e New York
- "… If we fail to meet the SLA guarantee in two consecutive months, we will automatically credit one day of the monthly fee for the service which has not been met…"

Quality of Web Service (QoWS)

- **Prestazioni**
  - Il 95% del tempo di risposta per determinati servizi e per determinati utenti non sarà inferiore ad un certo valore
- **Affidabilità**
  - Il sistema Web sarà funzionante per il 99.9[99]% del tempo
- **Sicurezza**
  - Il sistema garantirà il 100% della sicurezza per determinati servizi

Metriche per la QoS e QoWS

- Scelta una metrica (tempo di risposta, throughput, utilizzazione)
- Scelto un valore soglia X.

- **NO: media tra i valori campione osservati inferiore ad X.**

- **SI: 90 o 95-percentile dei valori campione osservati**

  Es., 95% dei tempi di risposta osservati deve essere inferiore ad X.

QoS *vs*. QoWS

"Less than 5 percent of organizations set and measure SLAs for distributed application availability and performance" (Gartner Group docs.)

"Network carriers do"

Perché?

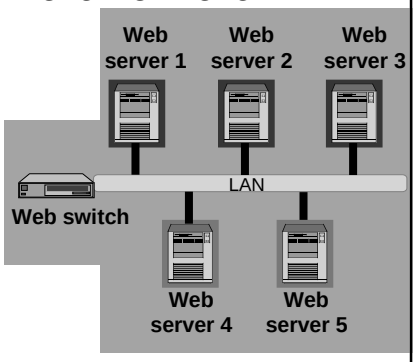
- **I network carrier controllano completamente le loro risorse (es., backbone)**

Perché è difficile far fronte alla QoWS

- Le architetture dei server Web devono far fronte ad enormi e spesso imprevedibili picchi di carico.
- Il Web cambia rapidamente: gli standard ed i protocolli sono ancora in evoluzione. **Ogni soluzione deve tener conto degli standard esistenti.**
- Internet non è (per ora) sotto il controllo di autorità centrali
- Possibili interventi (dipendenti dal ruolo aziendale)
  - Parti della rete (es., *backbone*)
  - Parti dell'infrastruttura (*sistemi di caching*)
  - Sito Web
  - **Mai sul client** (eccetto per Intranet)

I quattro principi della QoS applicati ai Web cluster

- **Classification**
  - utenti / servizi / effettuato a *livello4* o *livello7*
- **(Resource) Isolation**
  - partizionamento dei server
- **High utilization**
  - partizionamento dinamico
- **Access control (“declare”)**
  - effettuato dallo switch a livello7



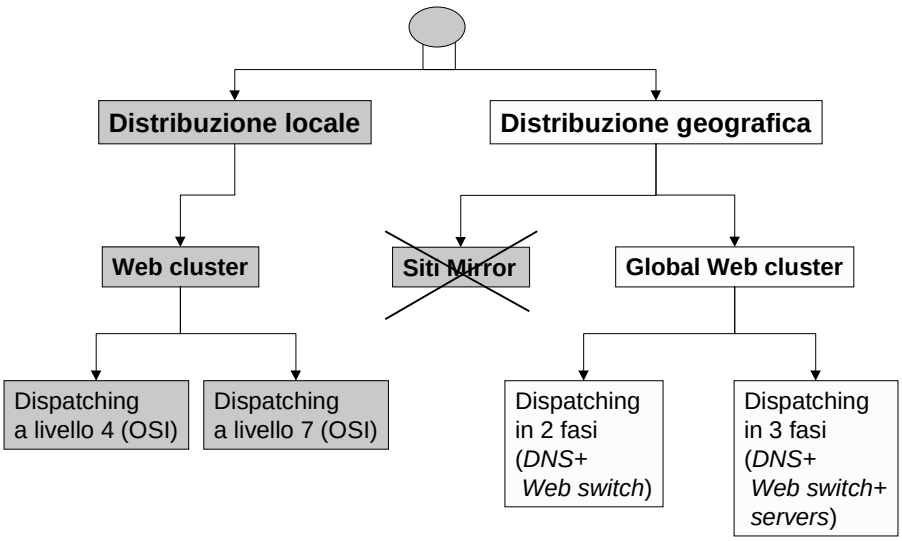
Architettura Web cluster per QoWS

- **Web switch a livello 7**
  - classificazione
  - controllo accessi
- **Partizionamento dinamico dei server** (*classe di politiche*)
- **Esperimenti preliminari:**
  - Soluzione che offre i migliori risultati
  - **Problema**: costo dell’analisi delle richieste a livello 7 rende il cluster poco scalabile!

Argomenti

0. Motivazioni e background
1. Sistemi Web ad alte prestazioni distribuiti localmente
2. Sistemi Web a Qualità del Servizio garantita (*cenni*)
3. **Sistemi Web ad alte prestazioni distribuiti geograficamente**

Sistemi con server Web multipli



Un esempio di *sito mirror*

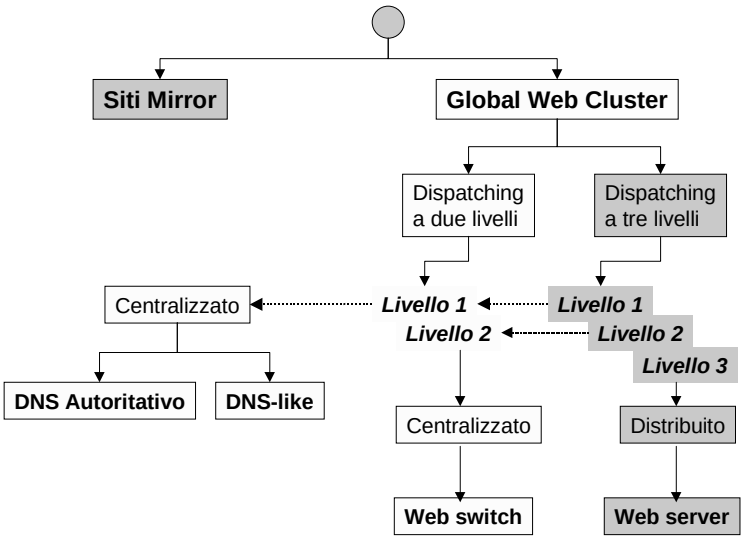
Mars Polar Lander Mission

Location of JPL Mirror Sites

Public Sector Mirror Sites

Location	Site Address	Load Capacity
SDSC - USA	http://mars.sdsc.edu	Bandwidth
Internet2 - USA	http://mars.dsi.internet2.edu	Bandwidth
NCSA - USA	http://www.ncsa.uiuc.edu/mars	Bandwidth
Mars Society - USA	http://missions.marssociety.org/mpl	Bandwidth
KSC - USA	http://www.ksc.nasa.gov/mars	Bandwidth
HIGP - USA	http://mars.pgd.hawaii.edu	Bandwidth

Sistemi Web distribuiti geograficamente

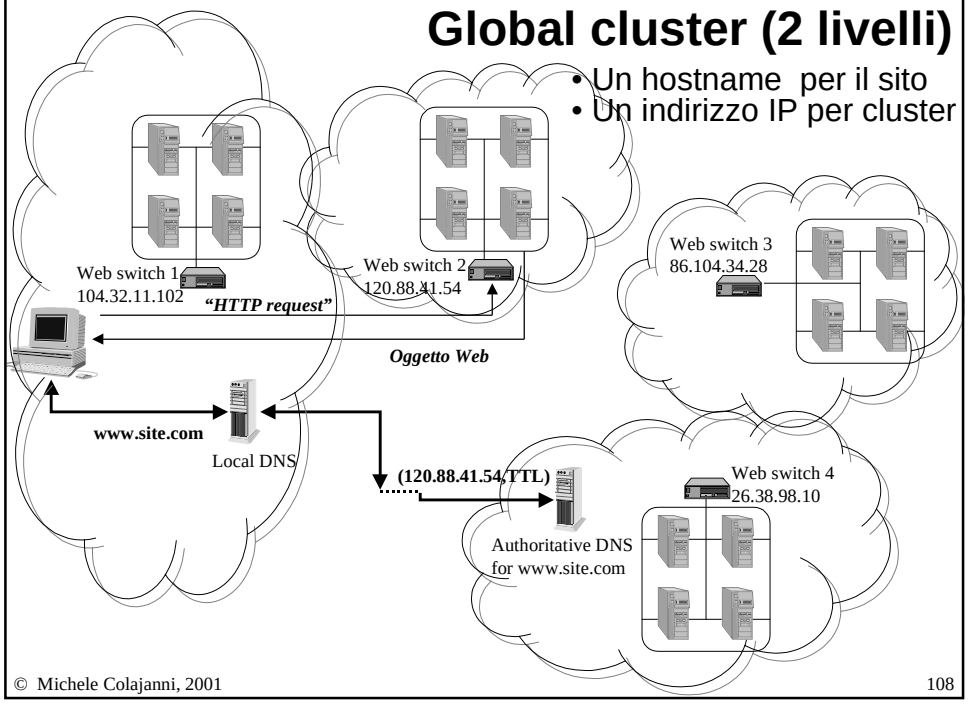


Global Web cluster

- *Sito Web implementato su di un'architettura di cluster distribuiti geograficamente tra diverse regioni Internet*
- **Indirizzi del sito Web**
  - Un solo hostname (es., "www.unimo.it")
  - Un indirizzo IP (*VIP dello switch*) per ogni cluster Web
- **Il DNS autoritativo del sito Web seleziona un cluster nella *fase di lookup* mediante:**
  - Round-robin
  - Prossimità geografica
  - Altro ...

Global cluster (2 livelli)

- Un hostname per il sito
- Un indirizzo IP per cluster



Dispatching di primo livello (mediante DNS)

- **Per il dispatching geografico si interviene tipicamente nella *fase di lookup*:**
  - il client richiede l'**indirizzo IP** del server corrispondente all'hostname indicato nell'URL
  - se l'hostname è valido, il client riceve la coppia

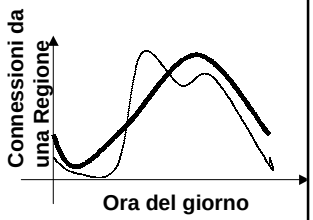
    < Indirizzo IP, Time-To-Live>
  - da:
    - cache indirizzi di qualche name server intermedio
    - **DNS autoritativo del sito** (opportunamente modificato) integrato o meno da altro componente, che può applicare diverse politiche di dispatching per selezionare il "cluster migliore"

Problemi del dispatching geografico

(di cui le politiche di dispatching devono tener conto)

Tipici problemi del dispatching Web

- Picchi di carico in alcune ore/giorni



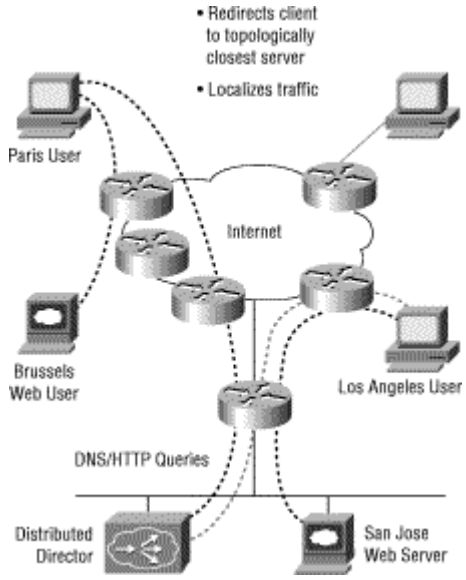
Problemi aggiuntivi

- Traffico dipendente dai fusi orari
- Distribuzione non uniforme dei client tra le regioni Internet
- Prossimità Internet tra client e server Web

- **(Per DNS)** Caching di [hostname-indirizzo IP] in name server intermedi per l'intervallo del Time-To-Live

Cisco Distributed Director

- **Entità centralizzata** che assegna ogni richiesta in base a:
  - **prossimità topologica** client-server (numero di hop ottenuto interrogando i router proprietari)
  - tempo di latenza client-server
  - carico sui server
- Duplice modalità di funzionamento:
  - **Domain Name System** (con TTL nullo): la locazione del client è stimata tramite l'indirizzo IP del name server locale del client
  - **Ridirezione HTTP**



Prossimità Internet

- La prossimità Internet è un interessante problema ancora aperto

  La prossimità geografica tra client e server *non implica prossimità Internet* (*round trip latency*)

  - **Valutazione *statica***
    - ♦ indirizzo IP del client per determinare la zona Internet (simile a distanza geografica)
    - ♦ numero di hop (informazione "stabile" più che "statica" [Pax97a])
      - *network* hops (e.g., `traceroute`)
      - *Autonomous System* hops (routing table queries)

      Non garantisce la selezione del miglior cluster Web, e.g., "*links are not created equal*"

Prossimità Internet (2)

– ***Valutazione dinamica della prossimità***
- round trip time (es., `ping`, `tcping` [Dyk00])
- bandwidth disponibile (es., `cprobe` [Car97])
- latenza delle richieste HTTP (es., *request emulation*)

Tempo aggiuntivo e costi di traffico per la valutazione

Un problema affine ancora aperto:
Correlazione tra il numero di hop e il round trip time?
- "Vecchie" misure: prossima a zero [Cro95]
- "Recenti" misure: elevata [McM99], mediamente elevata [Obr99]

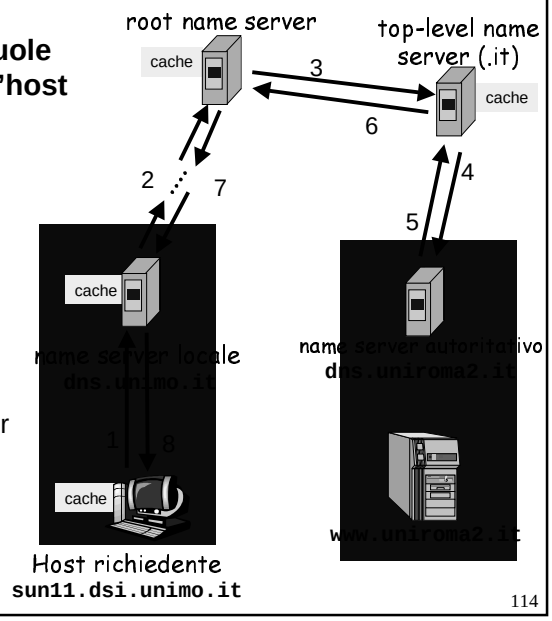
Sistema DNS (e caching indirizzi)

L'host sun11.dsi.unimo.it vuole conoscere l'indirizzo IP dell'host www.uniroma2.it

1) Contatta il suo *DNS locale*

2) Se necessario, il *DNS locale* può contattare altri DNS intermedi, ed eventualmente uno dei *root DNS*

3) Se necessario, il *root DNS* contatta il *DNS autoritativo* per quell'indirizzo (o un *top-level DNS* nel caso in cui non conosca un *DNS autoritativo* per quell'indirizzo)



Algoritmi di scheduling per DNS*

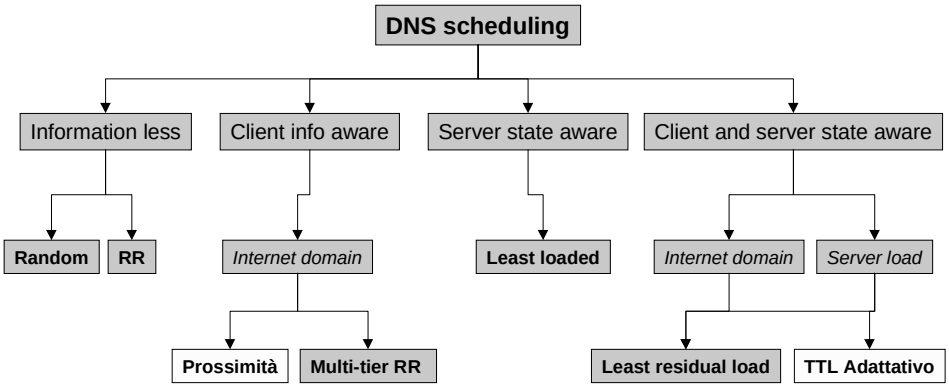
- **Statici**
  - Random
  - Round Robin
- **Client aware**
  - Prossimità al server Web
  - Dominio di provenienza
- **Server load aware**
- **Client and server aware**

* V. Cardellini, M. Colajanni, P.S. Yu, *IEEE Internet Computing*, May 1999

Azioni sul TTL

- TTL costante
  - Uso di TTL=0 per aumentare il controllo DNS [CisDD, Sch95, Bec98]
  - Problemi
    - DNS non-cooperativi
    - Cache indirizzi da parte dei browser
    - Rischi di sovraccarico del DNS autoritativo
- TTL Adattativo
  - Scelta del valore del TTL in modo adattativo a seconda della richiesta: si considera la popolarità del dominio del client e il carico dei server Web [Col98a]

Alfìgoritmi di dispatching del DNS*



* Classificazione e maggiori dettagli in [Col98b, Car99a]

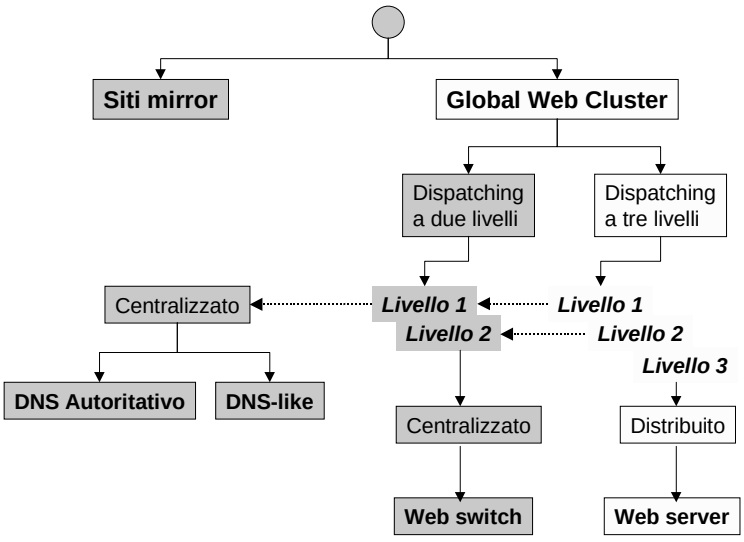
Problemi del dispatching DNS

- A causa del caching, nel caso di siti Web molto popolari, il DNS controlla solo il 5% del traffico in arrivo al sito
- A differenza del Web switch (che controlla il 100% del traffico in arrivo al sito), il DNS deve utilizzare algoritmi sofisticati (es., *TTL-adattativi*)
- Non sono stati trovati (*esistono?*) algoritmi di dispatching DNS in grado di evitare episodi di sovraccarico per tutte le classi di workload

Come risolvere i problemi del dispatching DNS

- **Fase di lookup**: integrare il dispatching DNS con un'altra entità
  - Es., *CiscoDistributedDirector*
- **Fase di richiesta**: integrare il dispatching DNS centralizzto con dispatching distribuito da parte dei server
  - **Ridirezione HTTP**
  - IP tunneling [Bes98, Lin]

Sistemi Web distribuiti geograficamente



Dispatching per Global Web Cluster

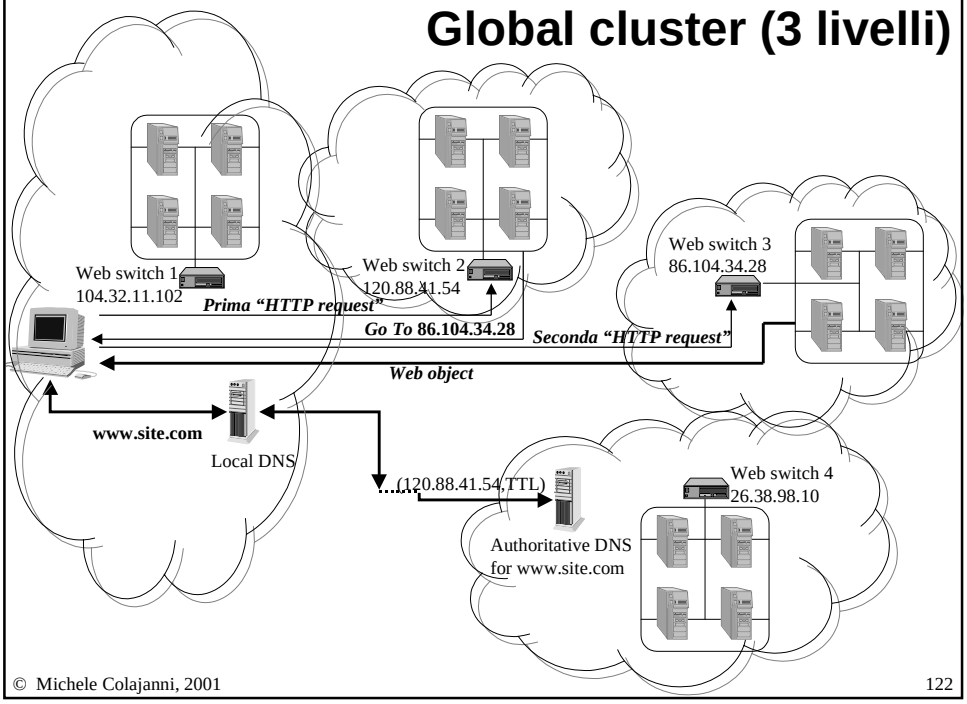
- **Indirizzi del sito Web**
  - Un hostname (e.g., “www.site.com”)
  - Un indirizzo IP per ogni Web cluster

Dispatching di primo livello
DNS autoritativo **seleziona il “miglior cluster”**

Dispatching di secondo livello
Web switch del **Web cluster seleziona il “miglior server”**

Dispatching di terzo livello
Ciascun Web server può ridirigere le richieste ad un altro cluster (es., mediante il meccanismo di ridirezione HTTP)

Global cluster (3 livelli)



Motivazioni per terzo livello di dispatching

Global Web cluster con due livelli di dispatching

- Controllo elevato sul carico che raggiunge il Web cluster *(buon bilanciamento intra-cluster)*
- Reazione lenta ad un cluster sovraccarico *(cattivo bilanciamento inter-cluster)*

Global Web cluster con tre livelli di dispatching:
Immediata reazione per spostare il carico da un Web cluster sovraccarico
(meglio "HTTP redirection" di "IP tunneling")

Ridirezione HTTP

- **Il meccanismo di ridirezione è parte del protocollo HTTP ed è supportato dagli attuali browser e client**
- **DNS e Web switch usano politiche di dispatching centralizzate**
- **La ridirezione è, invece, una politica di dispatching distribuita, in cui tutti i server Web possono partecipare al (ri-)assegnamento delle richieste**
- **La ridirezione è completamente trasparente per l'utente (non per il client!)**

message header
HTTP OK status code
302 - "Moved temporarily" to a new location

Ridirezione HTTP: pro e contro

PRO

- **La ridirezione HTTP è pienamente compatibile con tutti i client e server Web in quanto è implementata a livello applicativo**
- **La sua caratteristica distribuita soddisfa requisiti di affidabilità in quanto non introduce "singoli punti di guasto"**

CONTRO

- **Limita la possibilità di ridirigere solo richieste HTTP (in questo aspetto, l'***IP tunnelling* **è più generale)**
- **Aumenta il traffico in quanto ogni richiesta rediretta richiede una nuova connessione HTTP**

Tuttavia: la ridirezione **riduce** il tempo di risposta quando ***impatto del server > impatto della rete***

Global Web cluster

Two-levels dispatching	Three-levels dispatching
DNS+Web switch	*DNS+Web switch+Web servers*
• **Alteon WebSystems'** GSLB	• **RND Networks'** WSD-DS
• **CISCO's** DistributedDirector	• **Arrowpoint Comm.'s** Content Smart Redirect
• **Resonate's** Global Dispatcher	• **Radware** WSD-DS
• **F5 Networks'** 3DNS	• **Hermes** (*prototipo Geo4*)
• **HydraWeb Techs.'** HydraHydra	
• **Radware's** WSD-NP, WSD-DS	
• **Coyote Point Systems'** Envoy	
• **IBM** Network Dispatcher ISS	
• **Foundry Networks'** GSLB ServerIron	
• **Radware** WSD-NP	

Caratteristiche prototipo *Geo4* (1)

Tre livelli di assegnamento:

- **DNS** basato sulla prossimità
- **Web switch** effettua l'assegnamento della richiesta ai server del suo cluster mediante la politica *Weighted Round Robin* (WRR)
- **Ridirezione HTTP** effettuata dal singolo Web server verso un altro switch per risolvere situazioni (temporanee) di sovraccarico

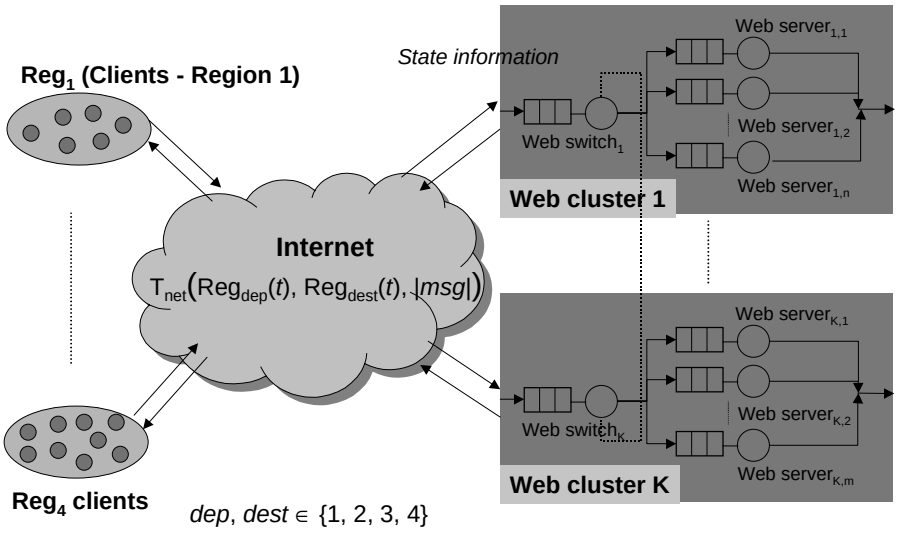
Caratteristiche prototipo *Geo4* (2)

- **Processo di ridirezione**
  - attivato da ciascun server in condizioni di carico critiche
  - la ridirezione aumenta il tempo di risposta e comporta un overhead sul server
- **Selezione oculata delle richieste da ridirigere**
  - ogni richiesta di risorsa (**all**)
  - dimensione della risorsa (**size**)
  - numero di oggetti che compongono la risorsa (**num**)

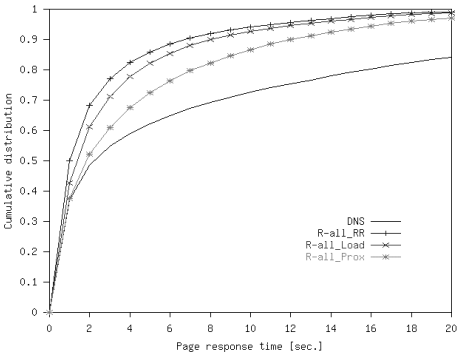
Studio politiche di ridirezione

- **Meccanismo di attivazione**
  - Distribuito: ciascun Web server, tipicamente quando sovraccarico
- **Politica di selezione (richieste da ridirigere)**
  - tutte le richieste (**All**)
  - tutte le richieste per risorse “grandi” (**Size**)
  - tutte le richieste per risorse con molti oggetti embedded (**Num**)
- **Politica di locazione (Web cluster a cui ridirigere le richieste)**
  - Round Robin (**RR**)
  - Funzione hash
  - Least loaded server (**Load**)
  - Prossimità client-server (**Prox**)

Modello del sistema *Geo4*

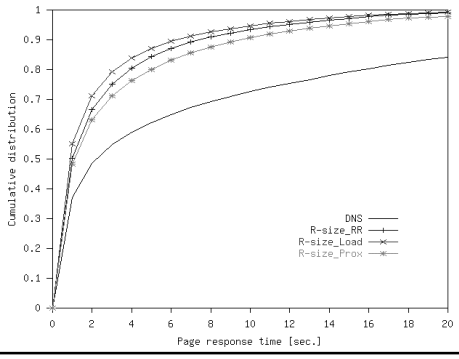


Analisi di prestazioni *Geo4*

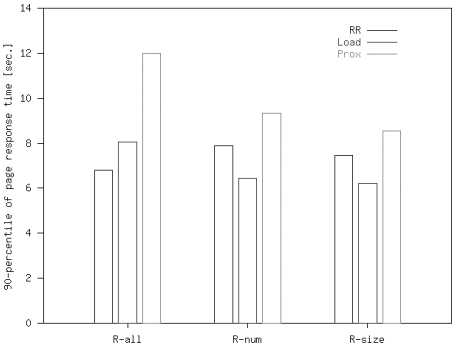


- *Politiche Redirect-all*
- *DNS usa prossimità*

- *Politiche Redirect-size*
- *DNS usa prossimità*

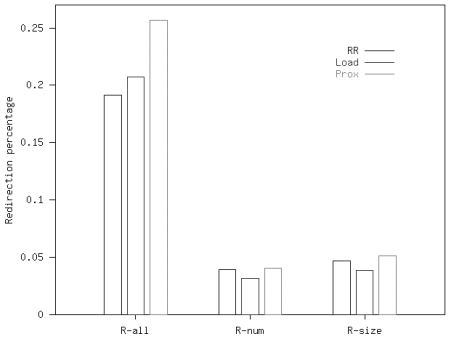


Analisi di prestazioni *Geo4* (2)

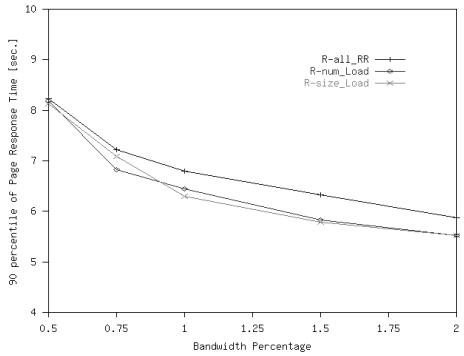


Tempo di risposta (90-percentile)

Percentuale di richieste ridirette

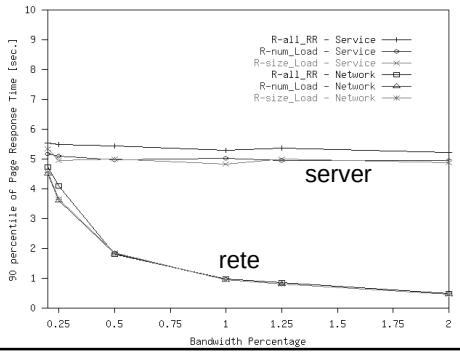


Analisi di sensibilità *Geo4* (*bandwidth*)



Tempo di risposta (90-percentile)

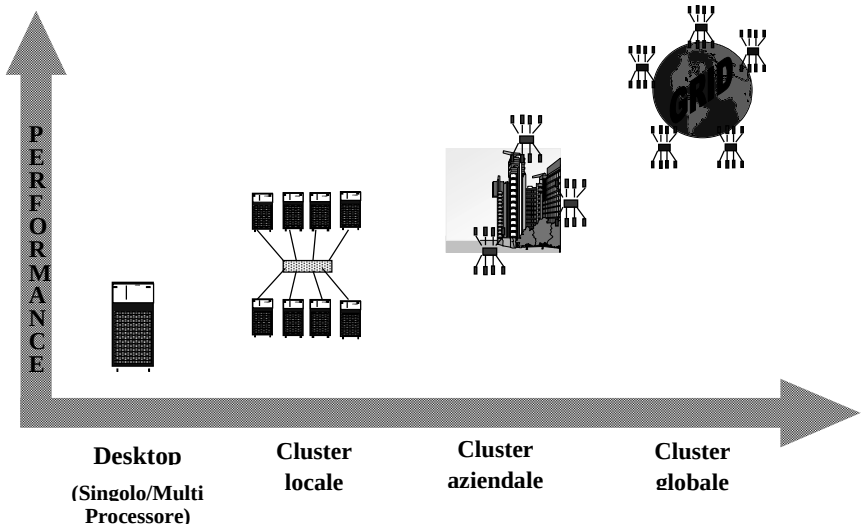
Contributi dovuti a server e rete



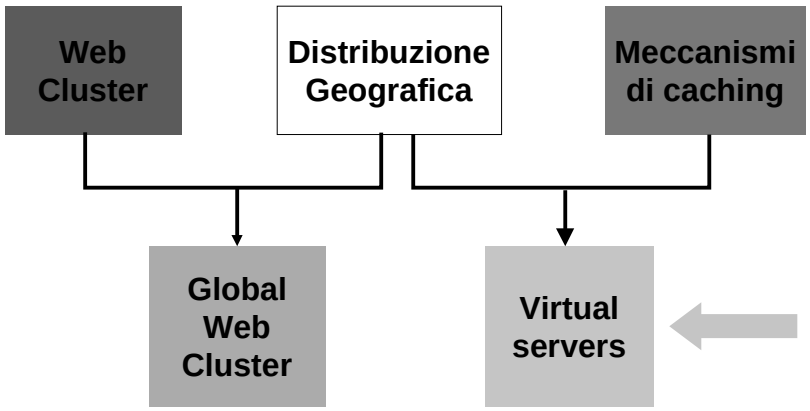
Uno sguardo al prossimo futuro ...

- Evoluzione piattaforme su scala geografica
- Evoluzione 1: Soluzioni integrate
- Evoluzione 2: Nuove applicazioni Web
- Evoluzione 3: Architetture cluster multi-tier
- Evoluzione 4: Universal Web

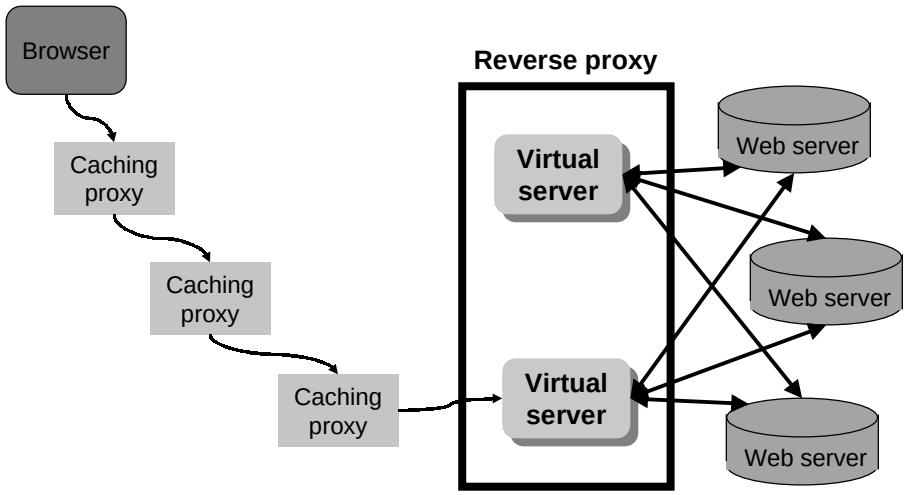
Evoluzione delle piattaforme (√)



Evoluzione 1: soluzioni integrate



Virtual server / Reverse proxy

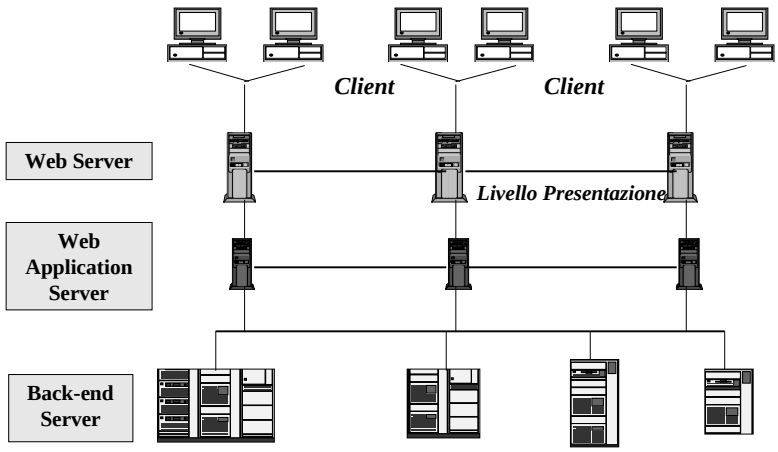


Evoluzione 2: Applicazioni Web

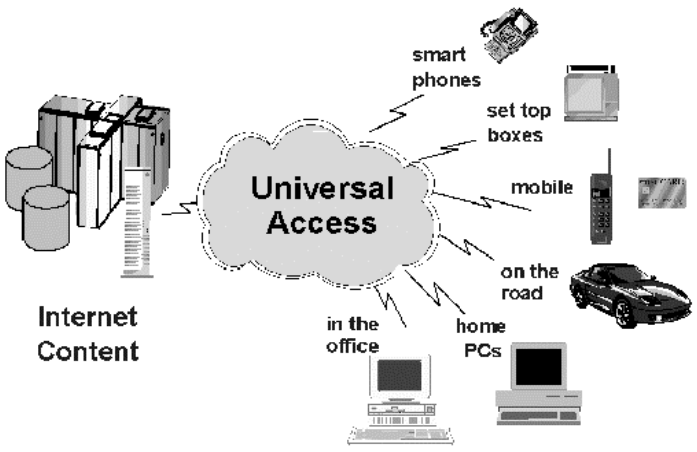
("*Main Web drivers*", TBL)

- Web publishing
  + prestazioni
- Electronic commerce
  + database + sicurezza + affidabilità + QoWS
- Education and training
  + streaming audio and video
- Universal Web
  + accessibilità + QoWS

Evoluzione 3: Architetture multi-tier



Evoluzione 4: Accesso universale al Web



Courtesy of IBM, 1999

Credits

WEB CLUSTER
Emiliano Casalicchio (dott. Univ. Roma Tor Vergata)
Mauro Andreolini (dott. Univ. Roma Tor Vergata)
Marco Emilio Poleggi (dott. Univ. Roma La Sapienza)

GLOBAL WEB CLUSTER & CACHING
Valeria Cardellini (assegn. Univ. Roma Tor Vergata)
IBM T.J. Watson Research Center, NY
Riccardo Lancellotti (dott. Univ. Modena)

QUALITY OF WEB SERVICES
Emiliano Casalicchio (dott. Univ. Roma Tor Vergata)
Marco Mambelli (dott. Univ. Modena)

UNIVERSAL WEB
Valeria Cardellini (assegn. Univ. Roma Tor Vergata)
IBM T.J. Watson Research Center, NY
Università Roma La Sapienza (prof. Bruno Ciciani)