HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Chapter 6: Distributed Systems: The Web

Fall 2011

Jussi Kangasharju

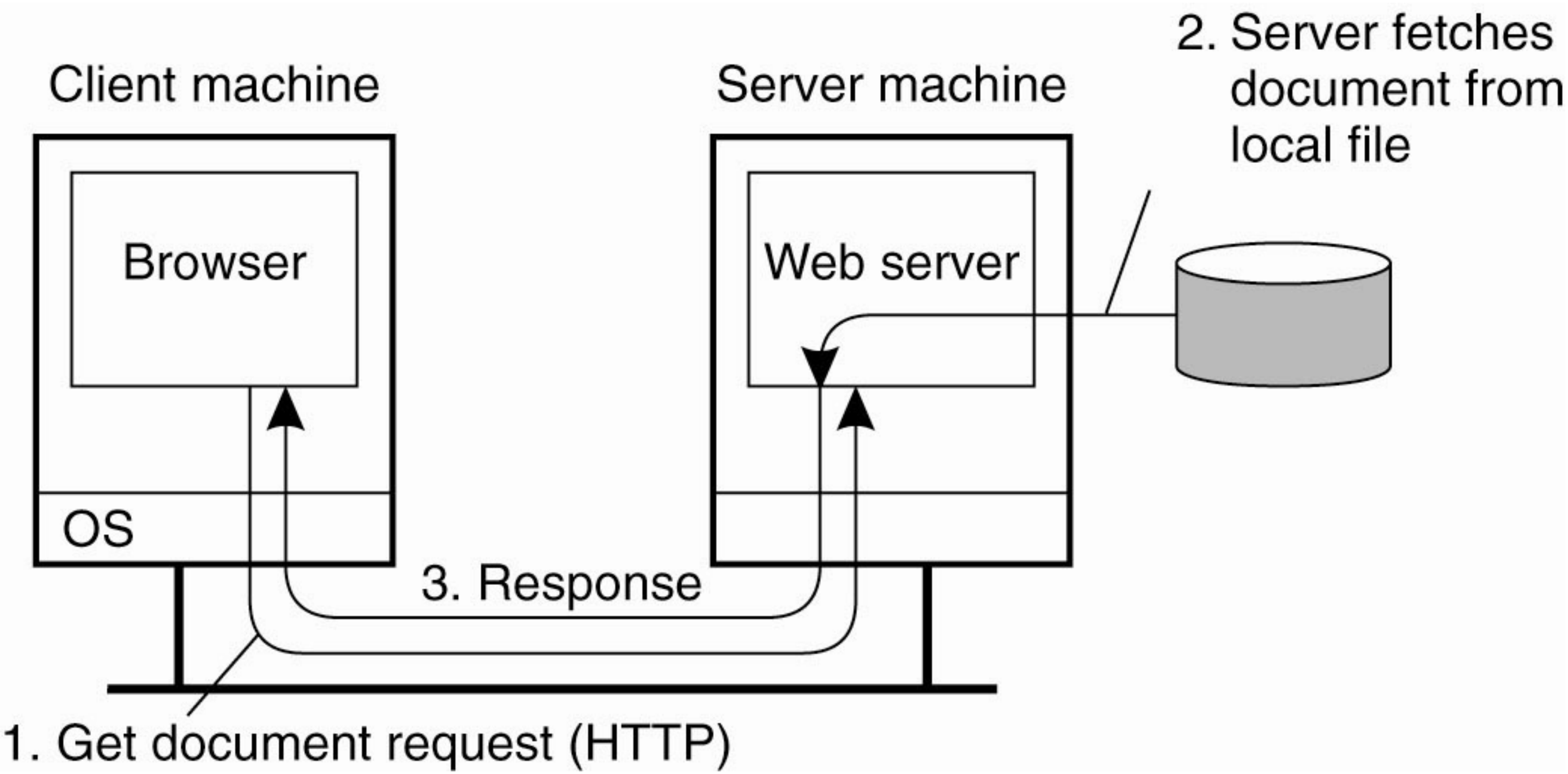
Chapter Outline

- Web as a distributed system
- Basic web architecture
- Content delivery networks
- Replication of web applications

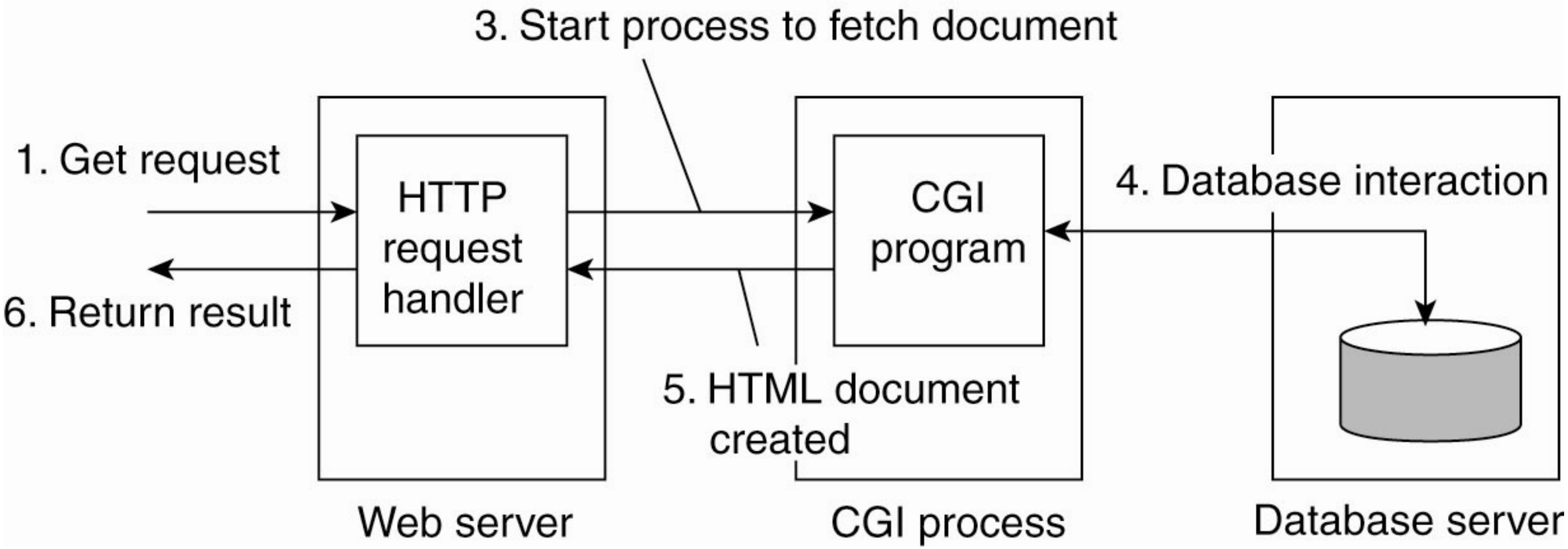
Web: Distributed or Not?

- Is the web a distributed system?
- Recall our definition:
  - Collection of independent computers → OK
  - Appears as single coherent system → ?!?
- Single coherent system = transparencies fulfilled?
- Sharing of resources → OK

Traditional Web-Based Systems


Client machine
Server machine
2. Server fetches document from local file
Browser
Web server
OS
3. Response
1. Get document request (HTTP)

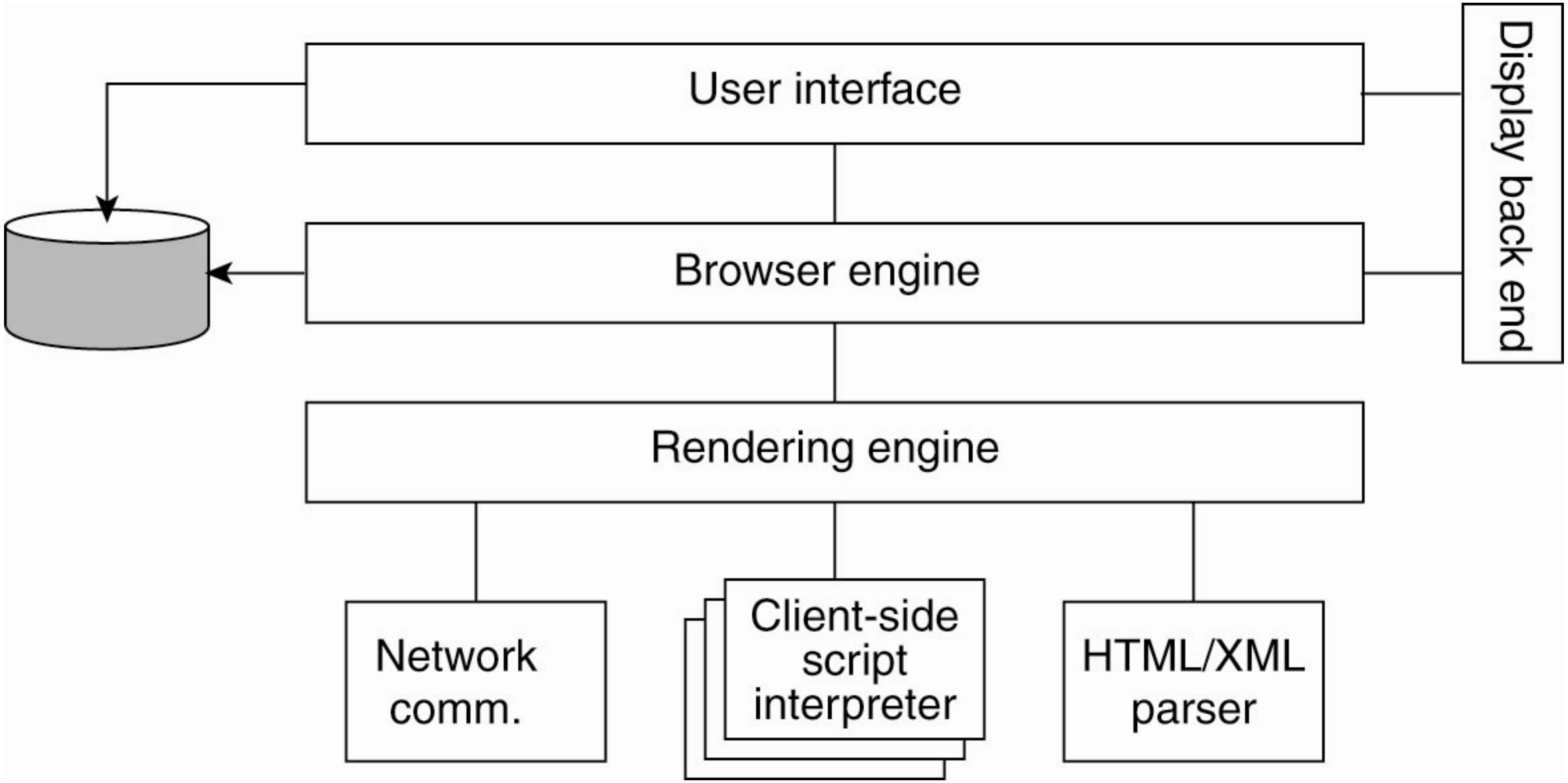
Multitiered Architectures



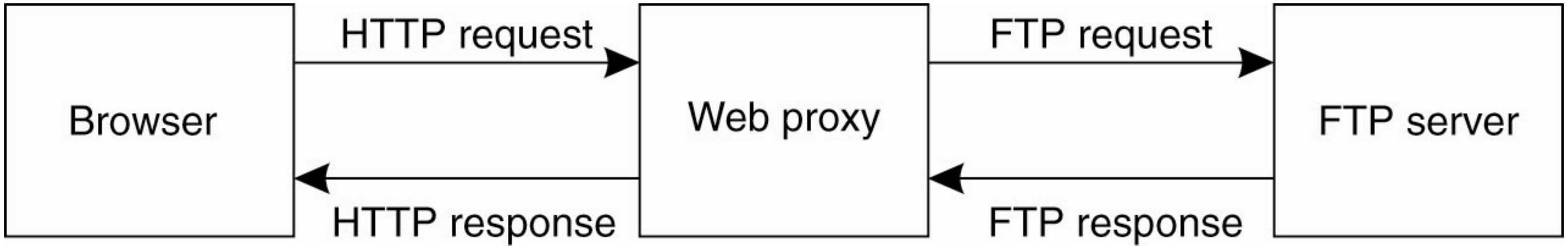
Important Elements

- Browser
- Servers and server farms
- Proxies
- Caching proxies

Processes – Clients (1)



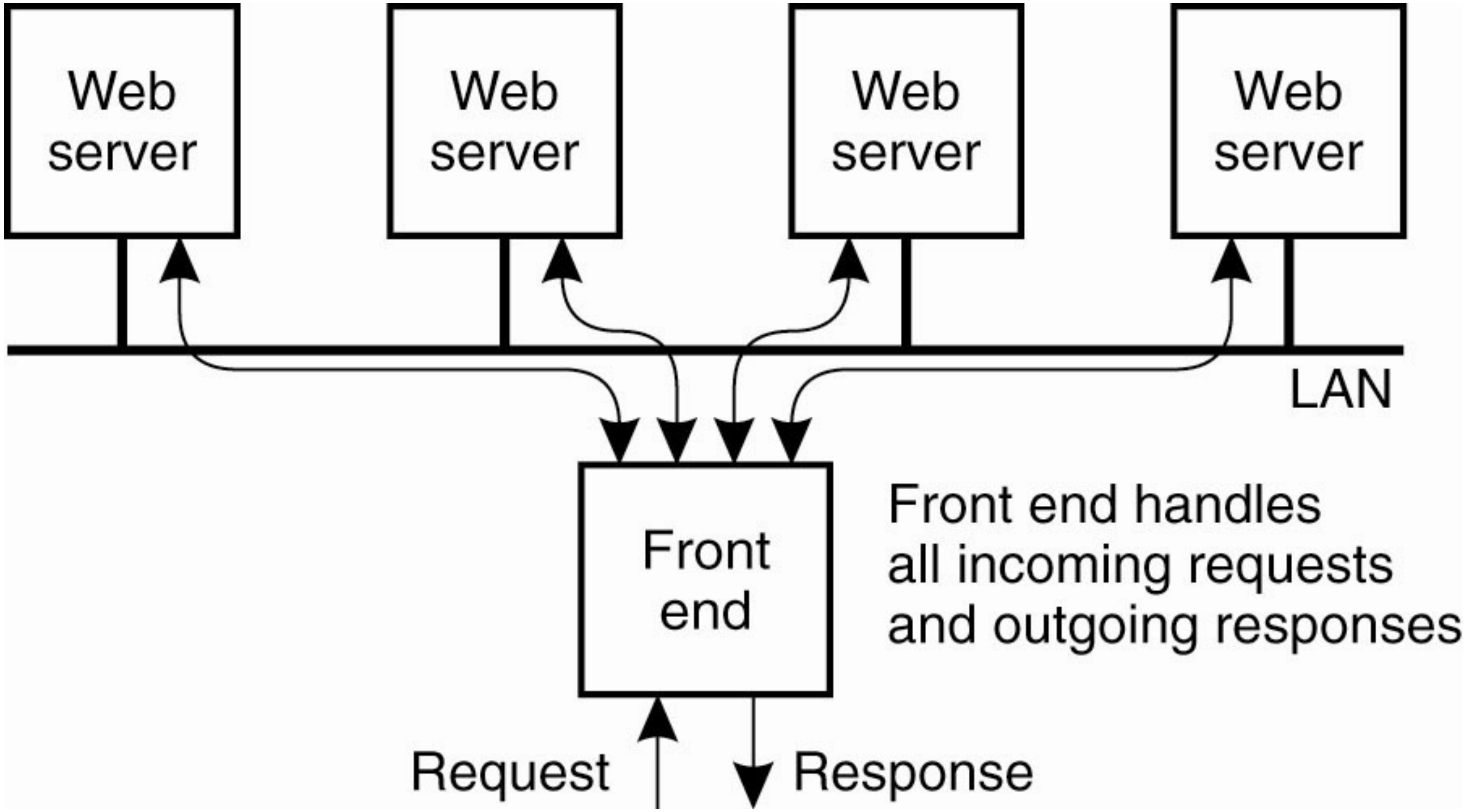
Processes – Clients With a Proxy



Terminology:
Proxy = simply proxying of requests and responses
Caching proxy = proxy with a cache

Commonly “proxy” = “caching proxy”

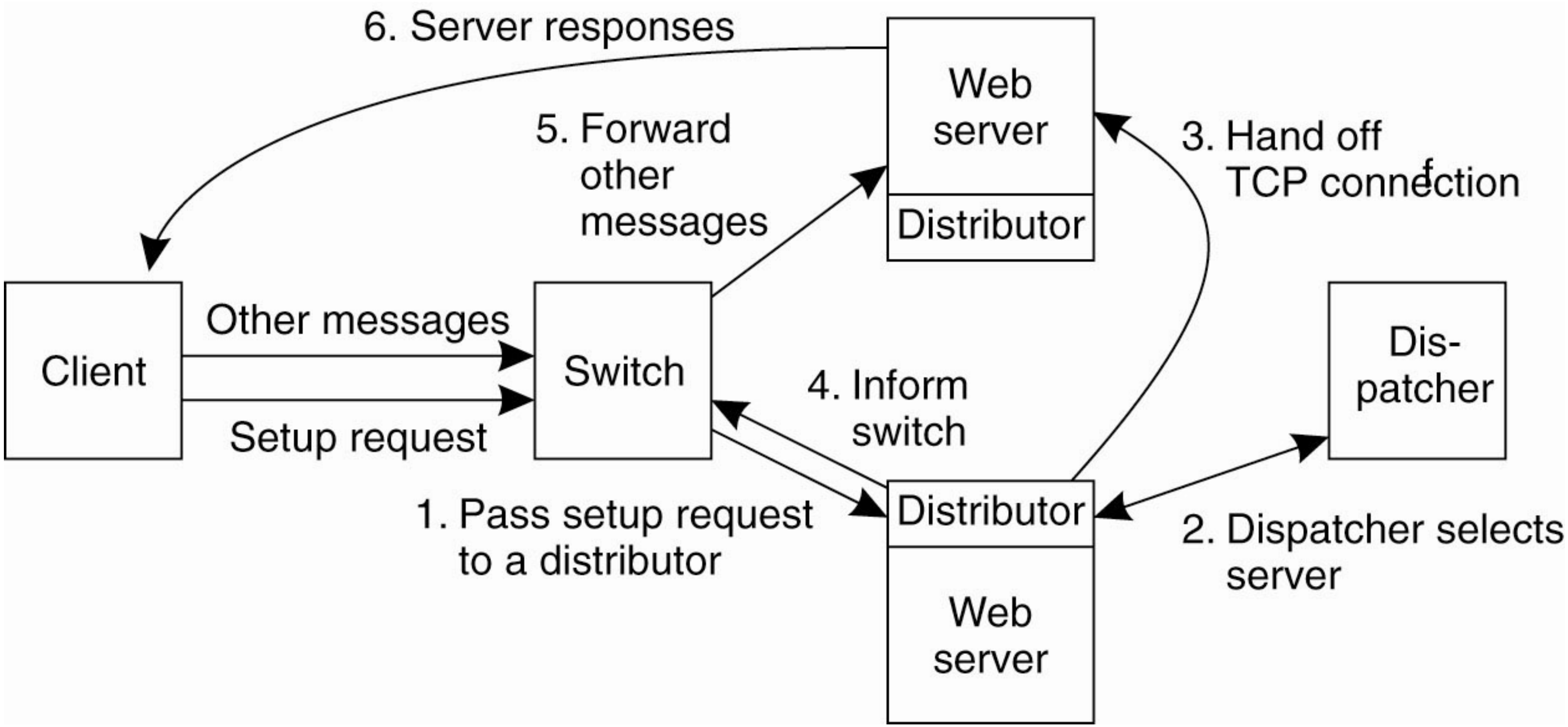
Web Server Clusters (1)



Front end handles all incoming requests and outgoing responses

Redirection independent of requested content

Web Server Clusters (2)

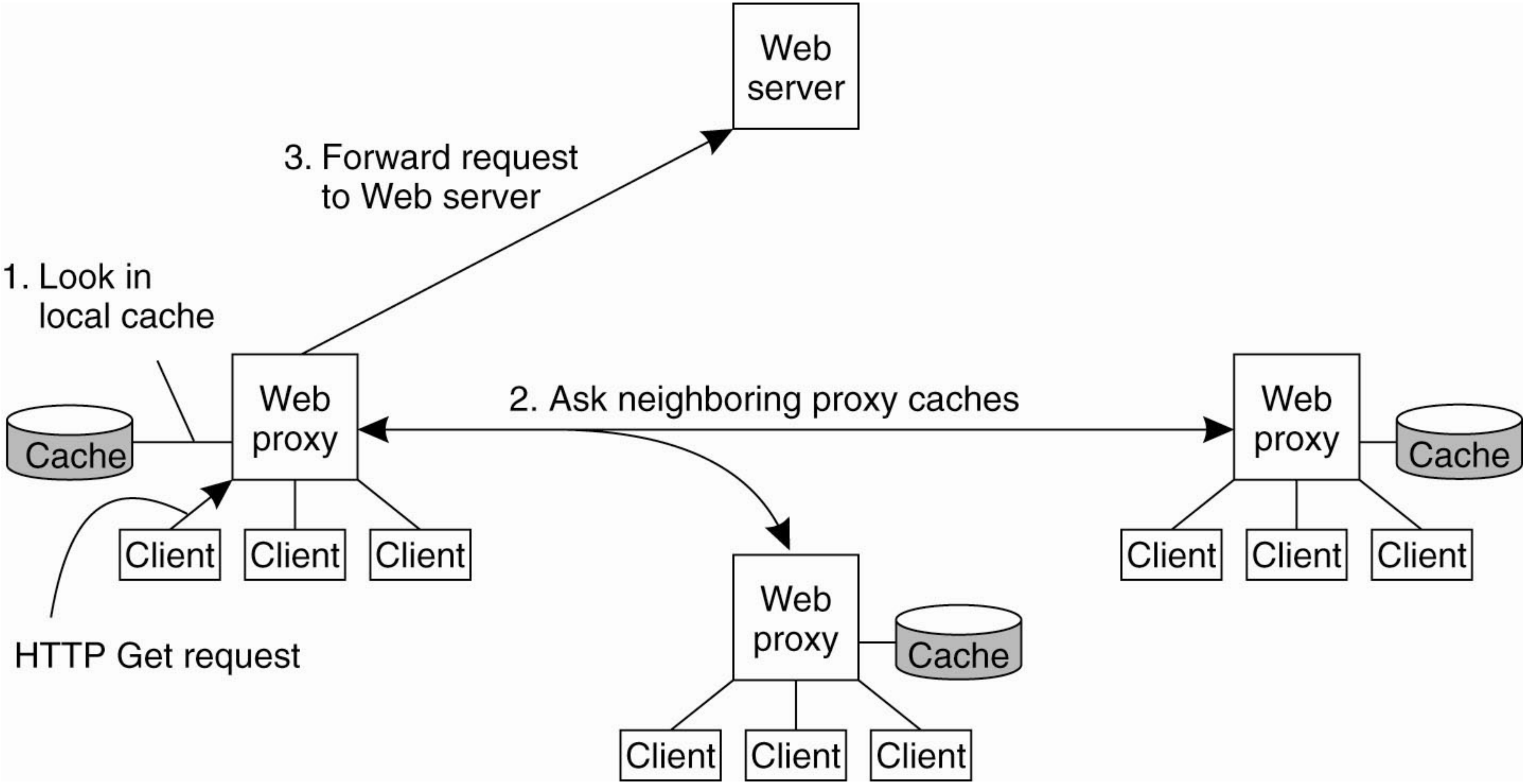


Redirection as function of requested content

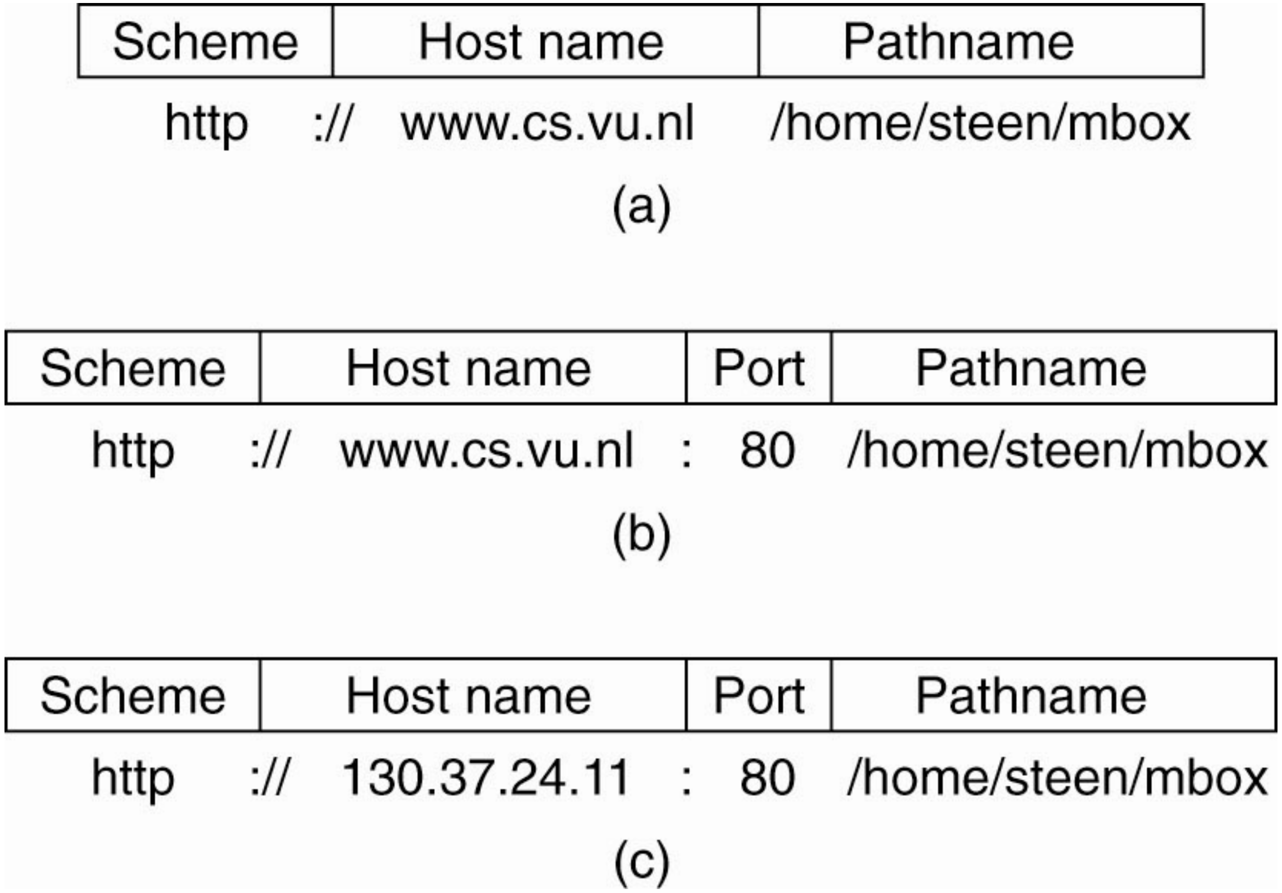
Content-aware vs. Content-agnostic

- Content-aware server selection:
  - Allows fine-grained selection and allocation of resources
  - Higher overhead at redirection point
  - No need to replicate all content on all servers
- Content-agnostic server selection:
  - Typically DNS load balancing
  - All servers must have identical content
  - Very high traffic → Even load distribution

Web Proxy Caching



Refresher: Names in the Web



Scheme	Host name	Pathname
http ://	www.cs.vu.nl	/home/steen/mbox

(a)

Scheme	Host name	Port	Pathname
http ://	www.cs.vu.nl :	80	/home/steen/mbox

(b)

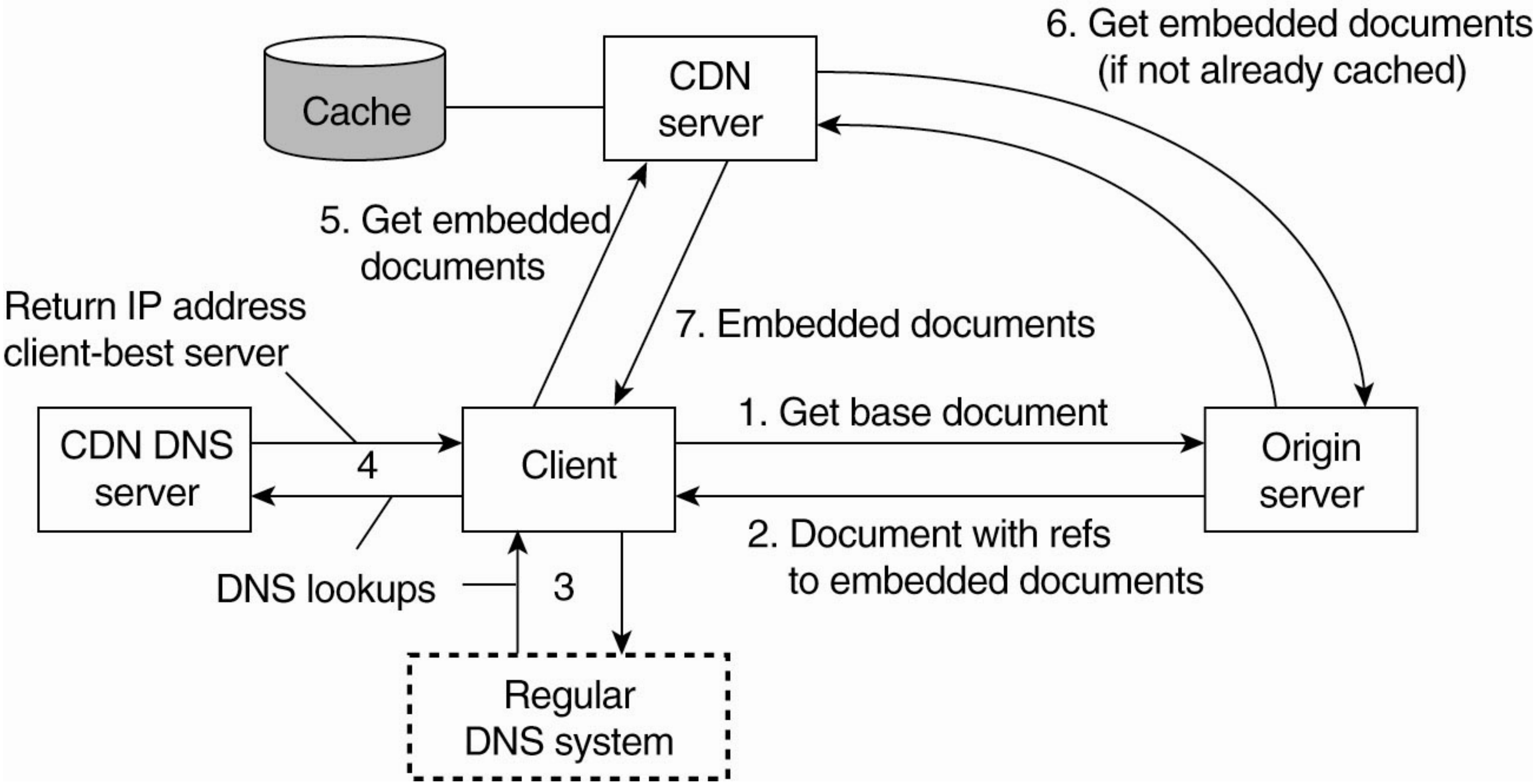
Scheme	Host name	Port	Pathname
http ://	130.37.24.11 :	80	/home/steen/mbox

(c)

Why Names Are Important?

- URLs identify content on the web
- URL typically identifies origin server by name → DNS
- Can do many tricks with DNS
- DNS load balancing for server farms
- DNS redirection for content delivery networks

Real CDN


6. Get embedded documents (if not already cached)
Cache
CDN server
5. Get embedded documents
Return IP address client-best server
7. Embedded documents
1. Get base document
CDN DNS server
4
Client
Origin server
2. Document with refs to embedded documents
DNS lookups
3
Regular DNS system

- The principal working of the Akamai CDN.

Total Redirection

- Any request for origin server is redirected to CDN
- CDN takes control of content provider’s DNS zone
- Benefit: All requests are automatically redirected
- Disadvantage: May send lots of traffic to CDN, hence expensive for the content provider

Selective Redirection

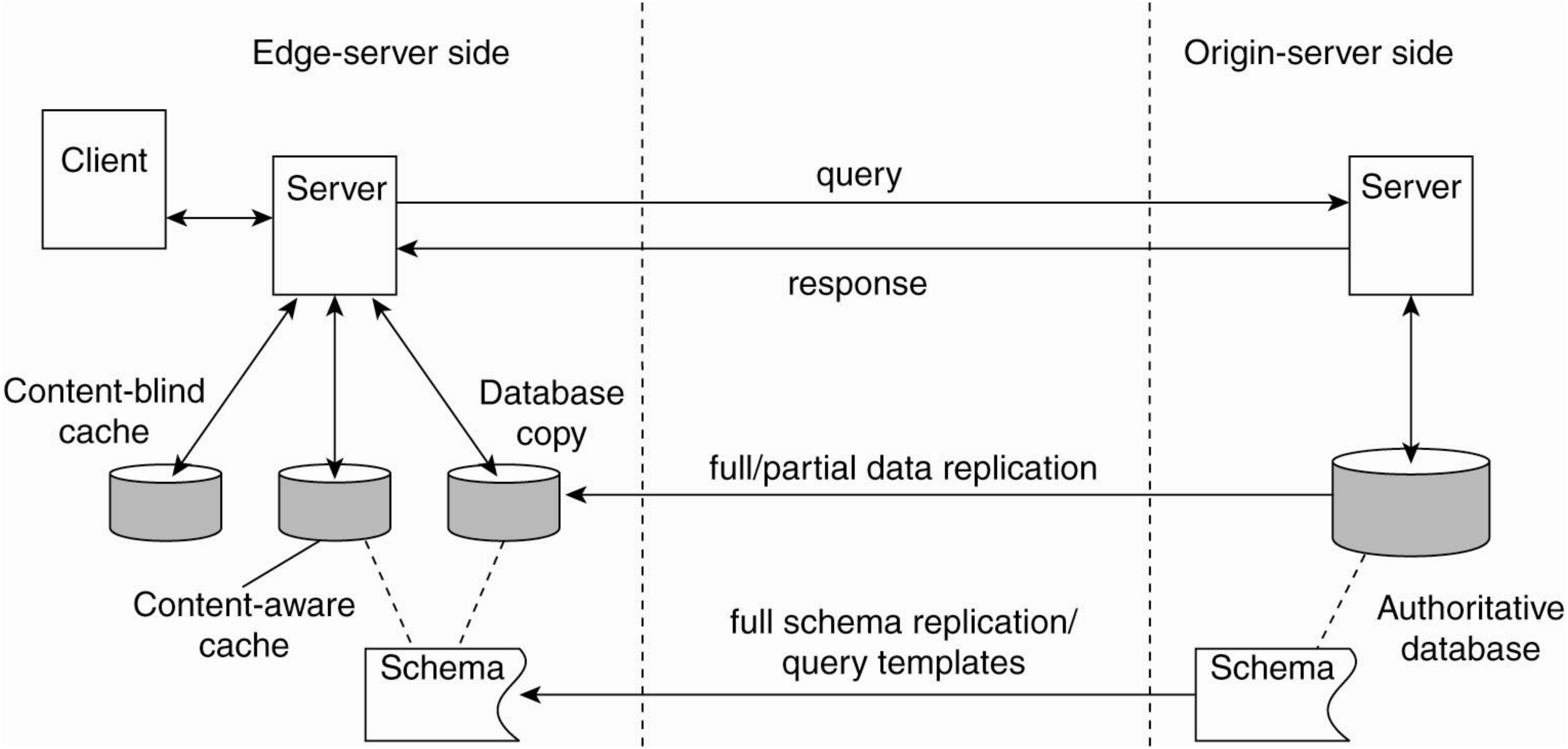
- Content provider marks which objects are to be served from CDN
  - Typically, larger objects like images are selected
- Refer to images as: <img src= http://cdn.com/foo/bar/img.gif>
- When client wants to retrieve image, DNS request for cdn.com gets resolved by CDN and image is fetched from the selected content server

- Pro: Fine-grained control over what gets delivered
- Con: Have to (manually) mark content for CDN

Replication of Applications

- Previously only replication of static content
- Can also replicate applications
- Recall: Application = Server + Database
- Replication of applications = Replication of database
- Full or partial replication of database?
  - Amount of data? Updates? Query containment?

Replication of Applications



Chapter Summary

- Web as a distributed system
- Basic web architecture
- Content delivery networks
- Replication of web applications