

Part 12

殷亚凤

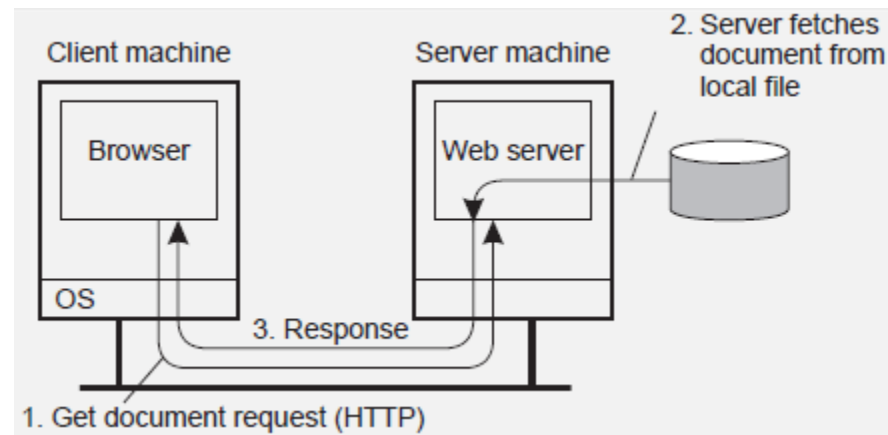
Email: yafeng@nju.edu.cn

Homepage: <http://cs.nju.edu.cn/yafeng/>

Room 301, Building of Computer Science and Technology

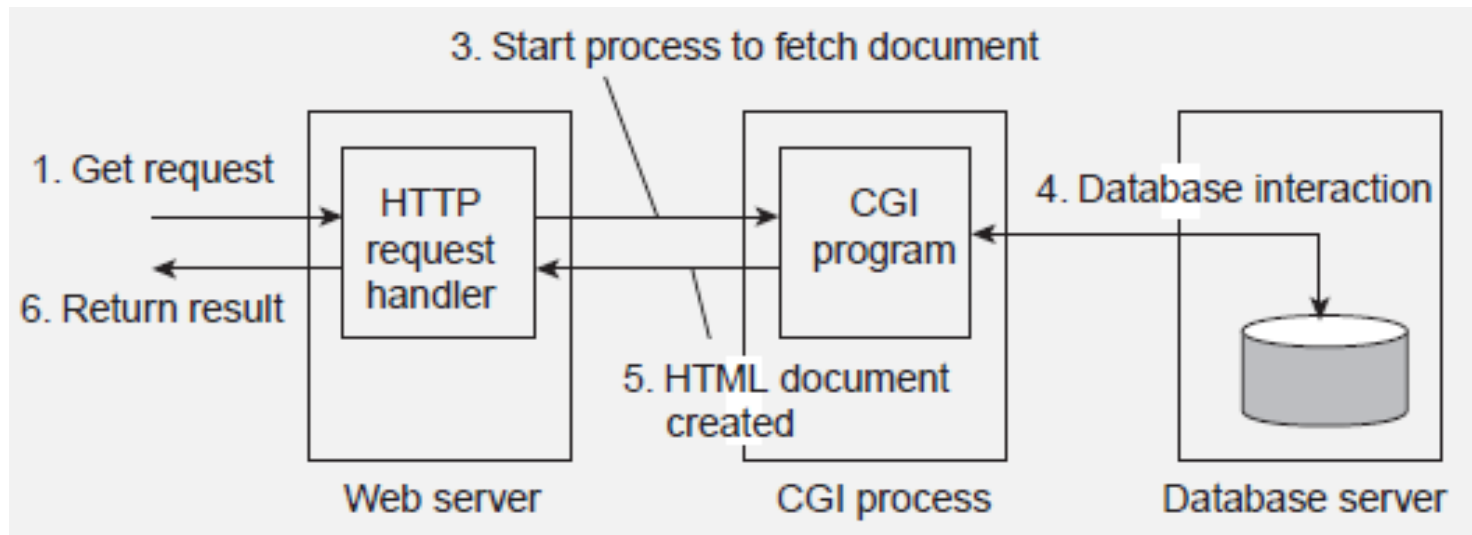
Distributed Web-based systems

- The WWW is a **huge client-server system** with millions of servers; each server hosting thousands of **hyperlinked documents**.
- Documents are often represented in **text** (plain text, HTML, XML)
- Alternative types: images, audio, video, applications (PDF, PS)
- Documents may contain **scripts**, executed by client-side software



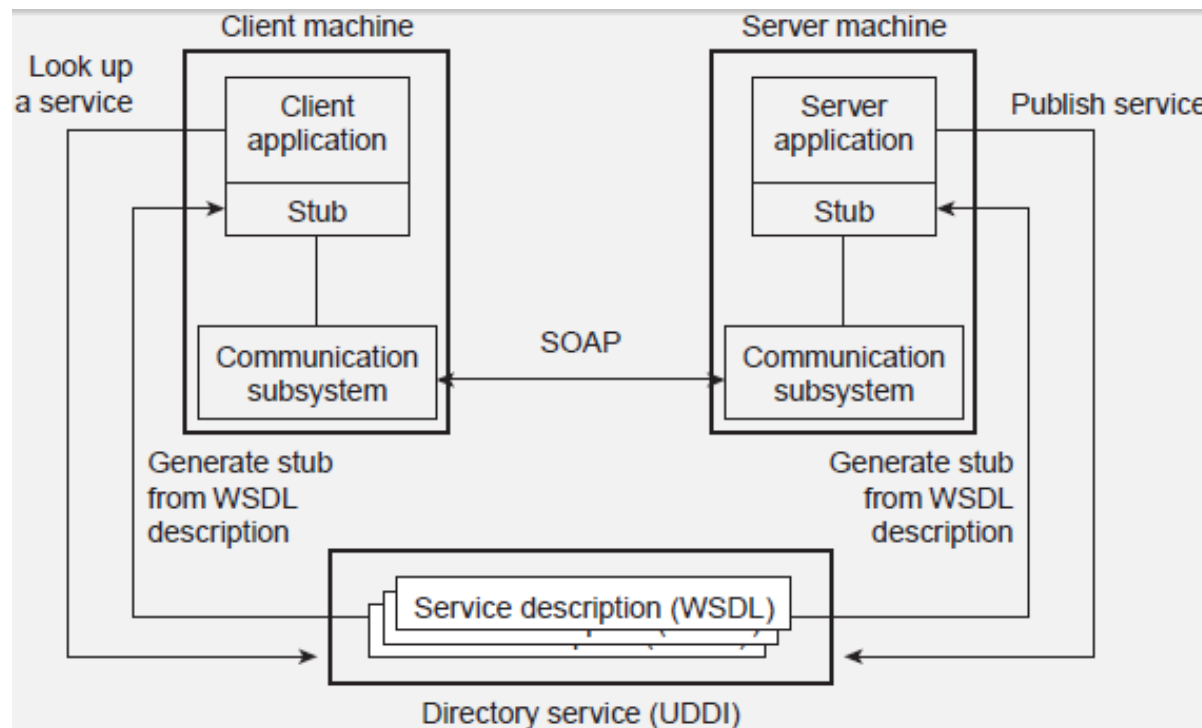
Multi-tiered architectures

- Already very soon, **Web sites** were organized into **three tiers**.



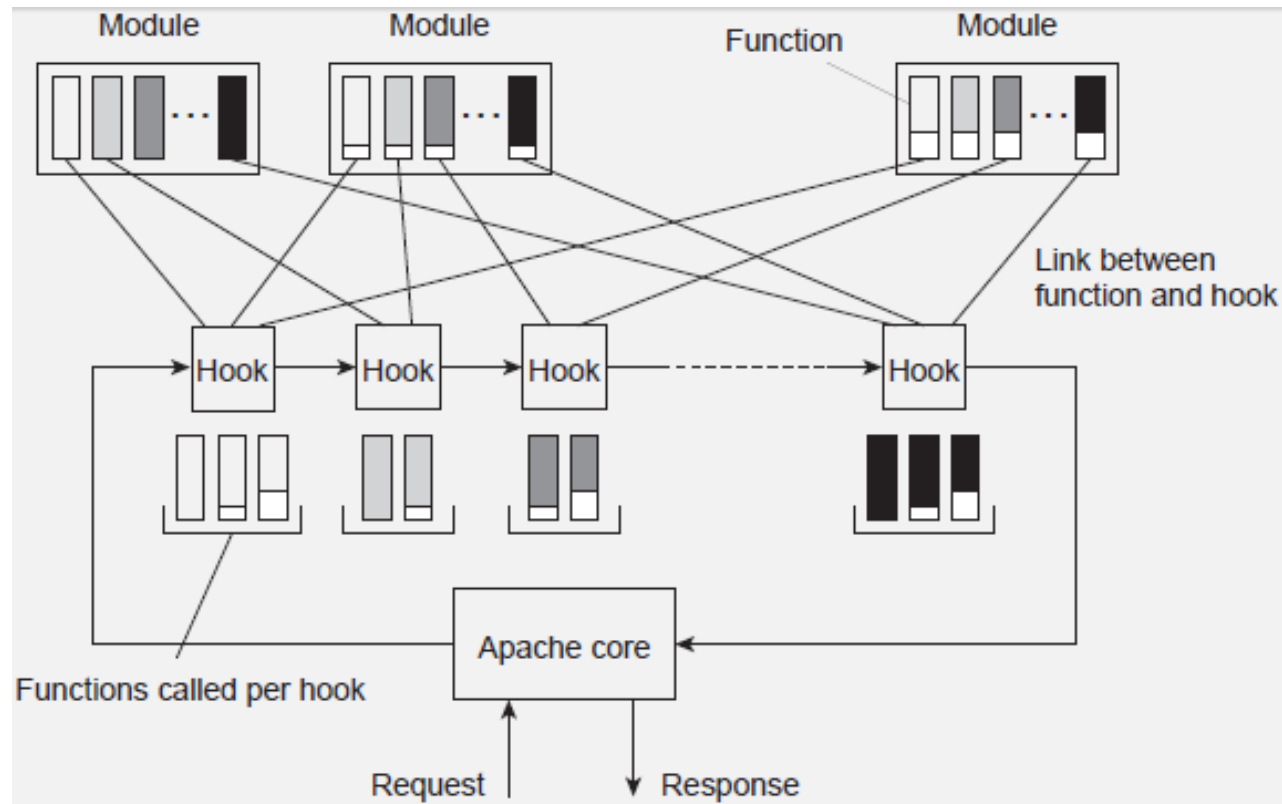
Web services

- At a certain point, people started recognizing that it is was more than just user \leftrightarrow site interaction: **sites could offer services to other sites** \Rightarrow standardization is then badly needed.



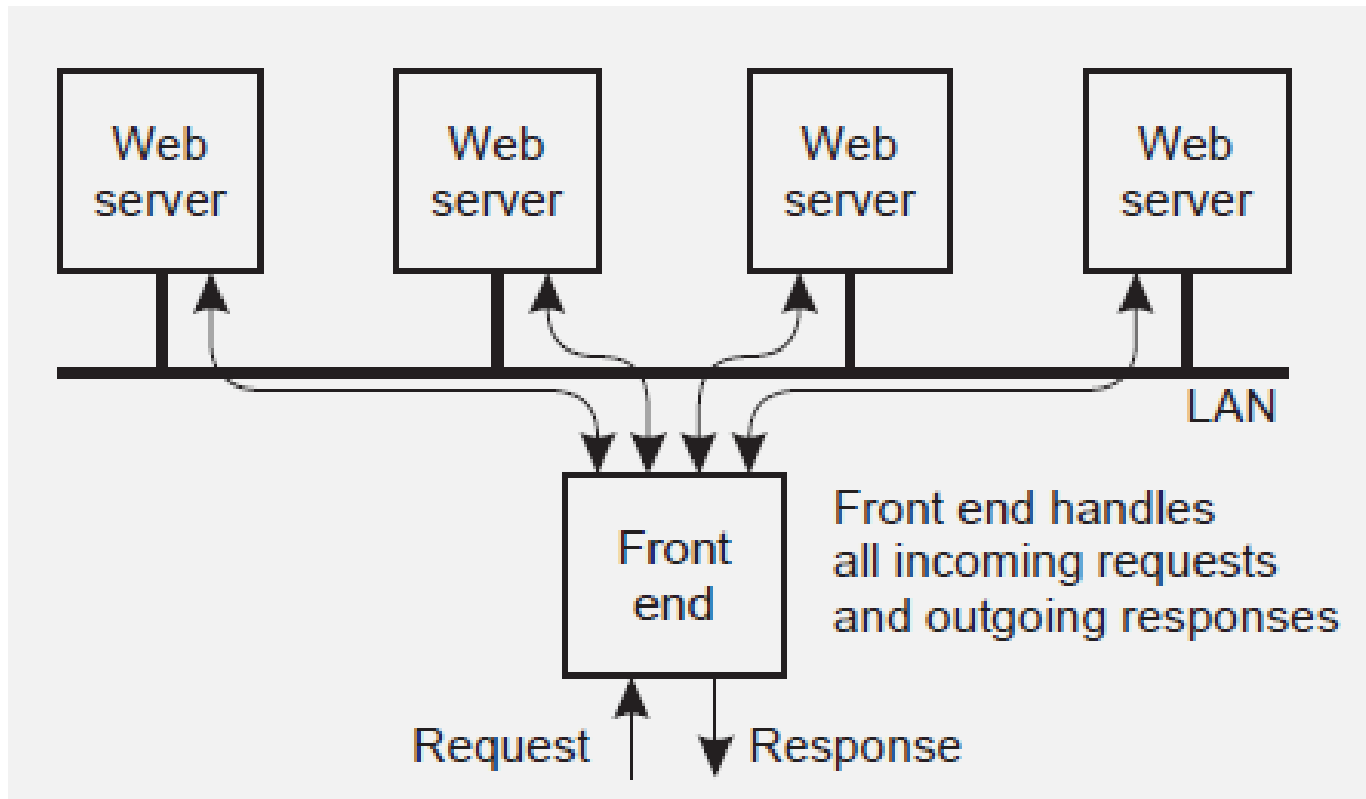
Apache Web server

- The server is internally organized more or less according to the steps needed to **process an HTTP request**.



Server clusters

- To improve performance and availability, WWW servers are often clustered in a way that is transparent to clients.

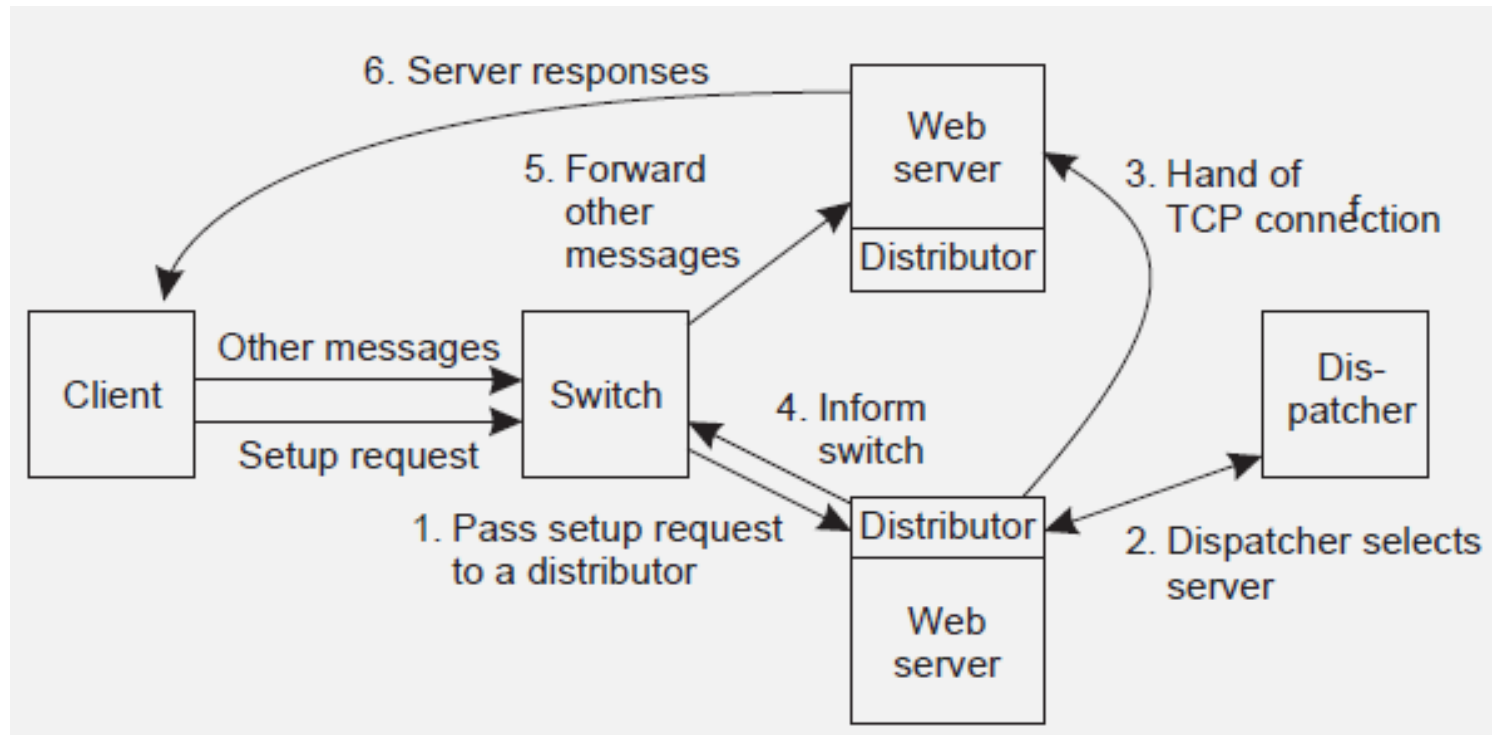


Server clusters

- Problem: The **front end** may easily get **overloaded**, so that special measures need to be taken.
 - **Transport-layer switching**: Front end simply passes the TCP request to one of the servers, taking some performance metric into account.
 - **Content-aware distribution**: Front end reads the content of the HTTP request and then selects the best server.

Server Clusters

- Question: Why can **content-aware distribution** be so much better?



Web proxy caching

- Sites install a separate proxy server that handles all outgoing requests. Proxies subsequently cache incoming documents.

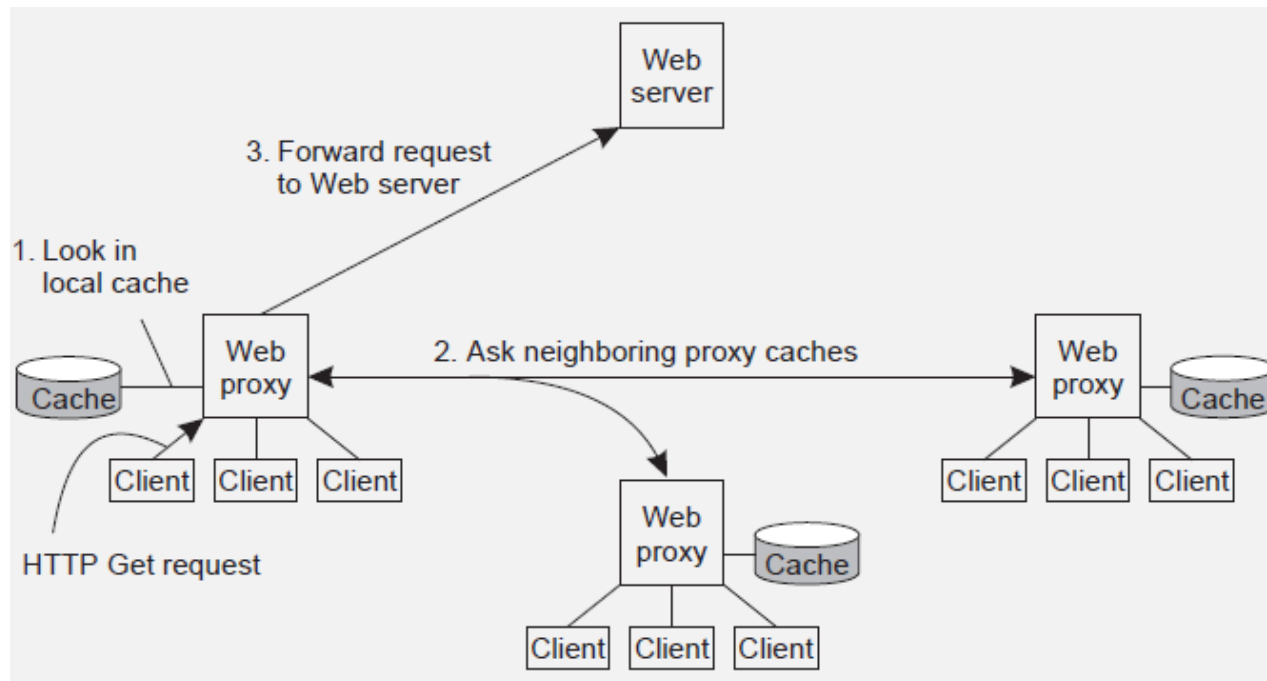
Cache-consistency protocols:

- Always verify validity by contacting server
- Age-based consistency:

$$T_{expire} = \alpha \cdot (T_{cached} - T_{last_modified}) + T_{cached}$$

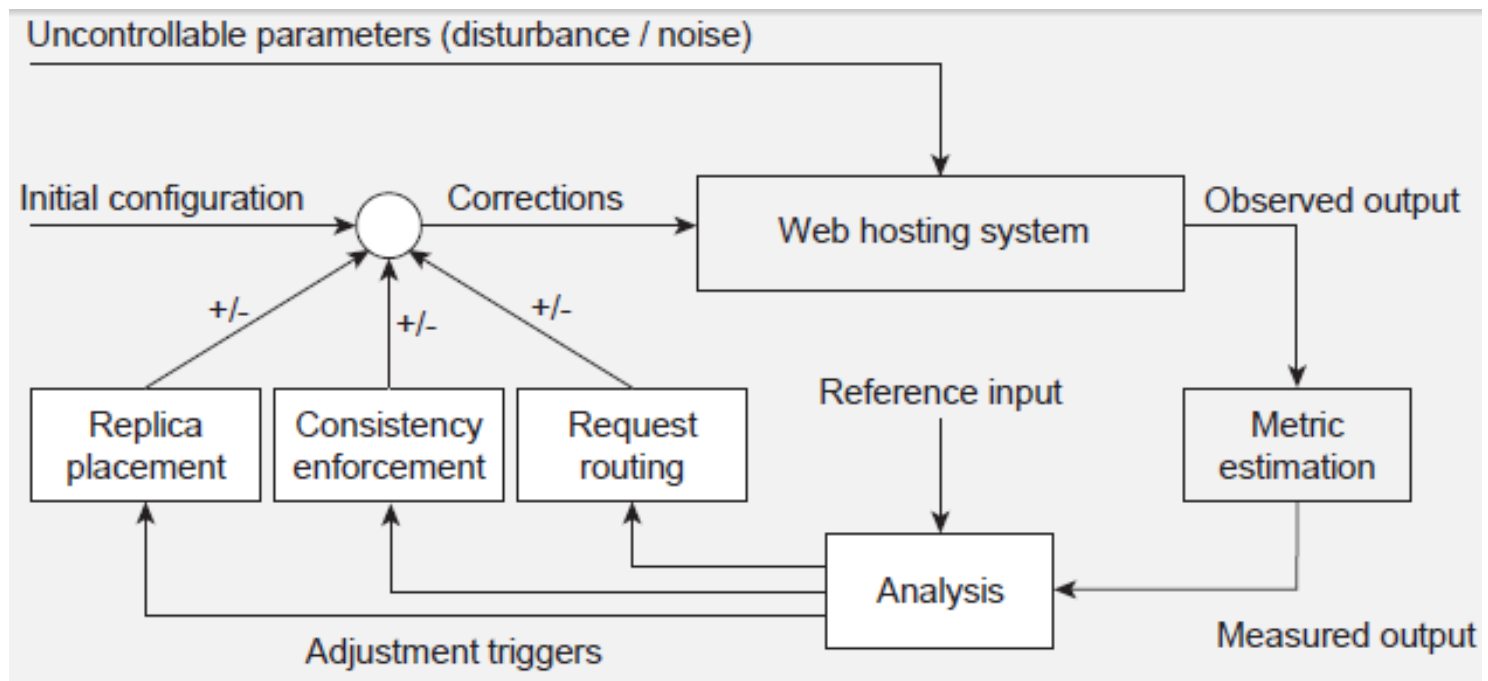
Web proxy caching

- Basic idea(cnt'd): **Cooperative caching**, by which you first check your neighbors on a cache miss



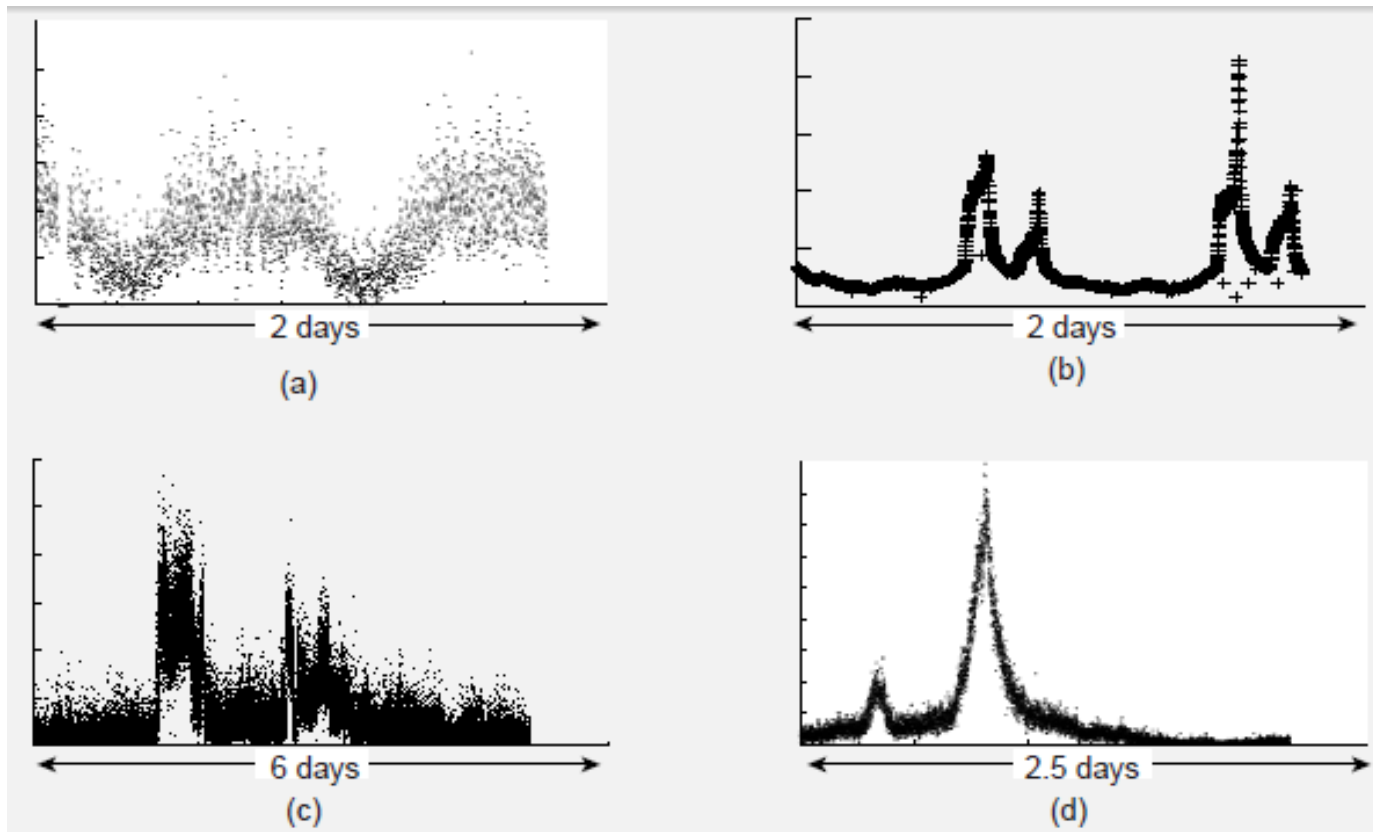
Replication in Web hosting systems

- By-and-large, [Web hosting systems](#) are adopting replication to increase performance. Much research is done to improve their organization. Follows the lines of [self-managing systems](#).



Handling flash crowds

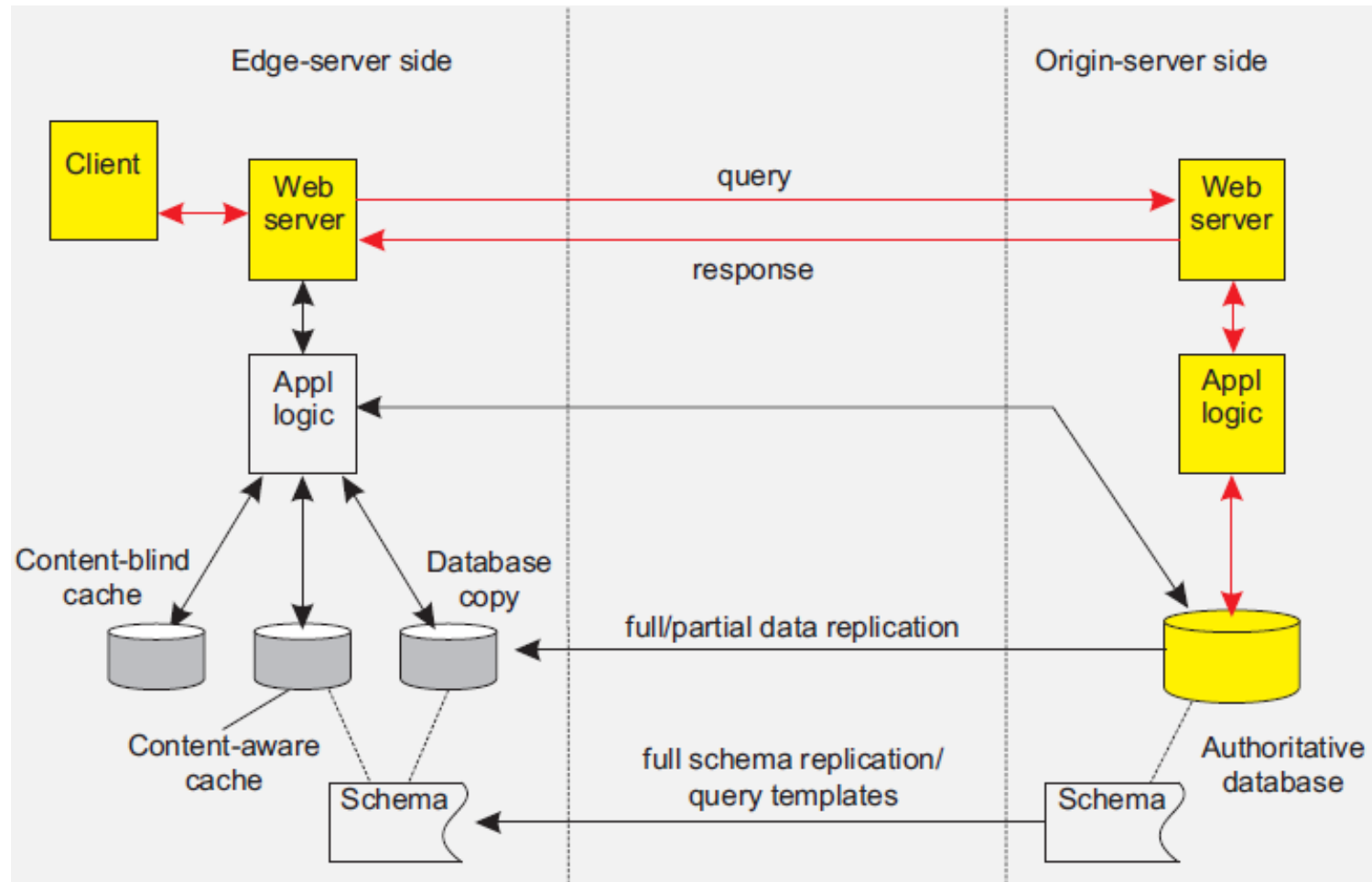
- We need **dynamic adjustment** to balance resource usage. Flash crowds introduce a serious problem.



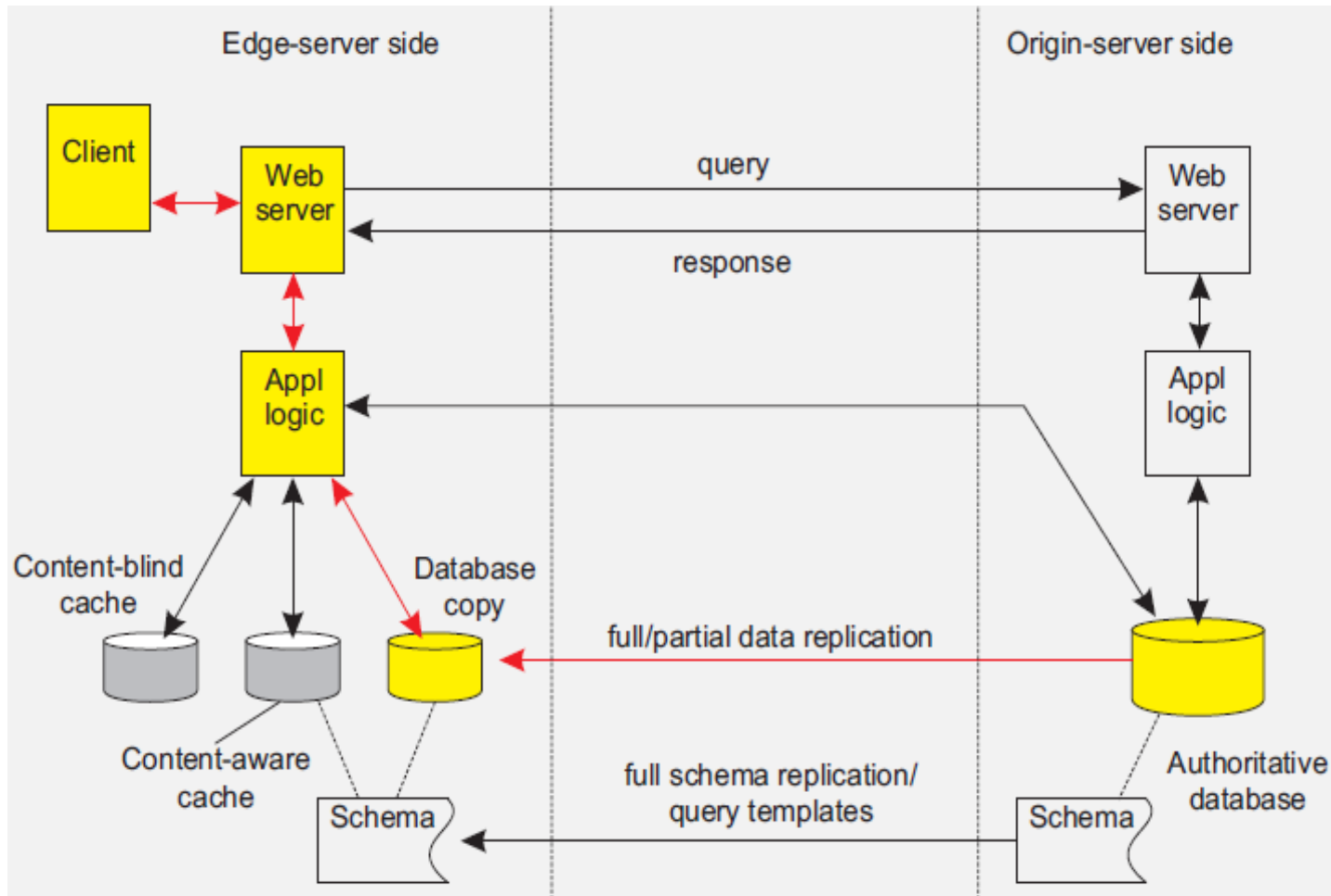
Replication of Web applications

- Replication becomes more difficult when dealing with databases and such. No single best solution.
- Assumption: Updates are carried out at origin server, and propagated to edge servers.

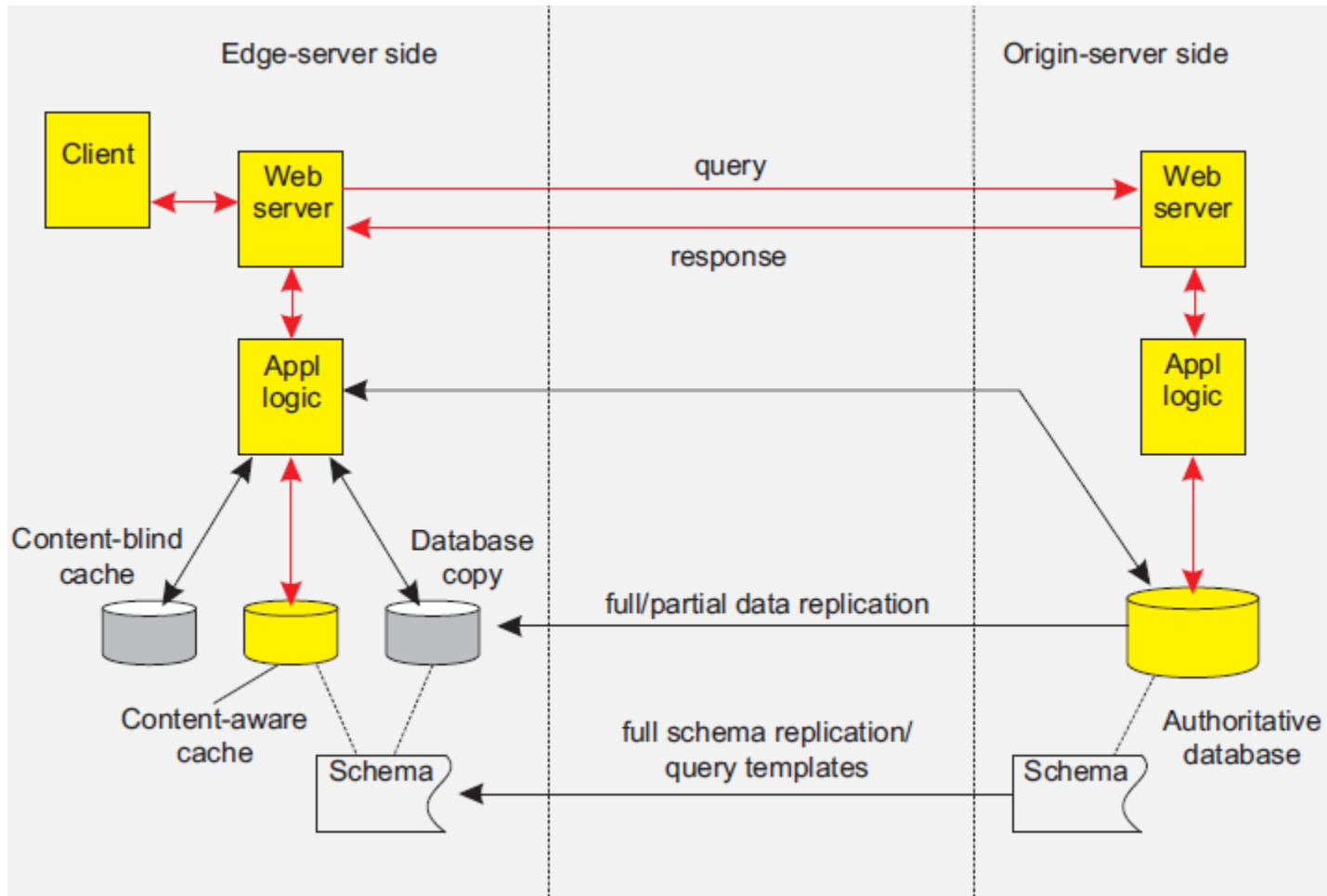
Replication of Web applications: normal



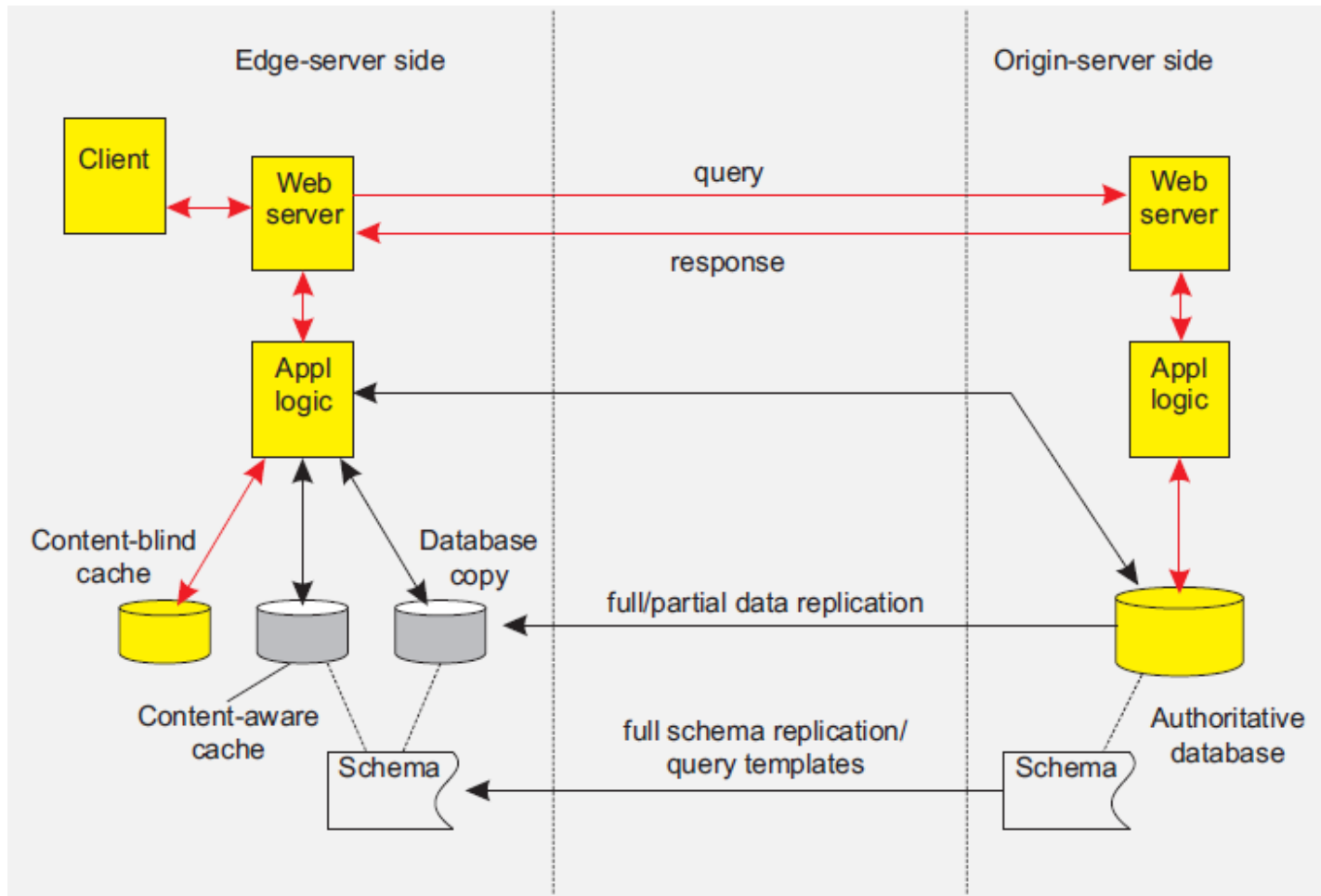
Replication Web apps.: full/partial replication



Replication Web apps.: content-aware caching



Replication Web apps.: content-blind caching



Replication of Web applications

- **Full replication**: high read/write ratio, often in combination with **complex queries**.
- **Partial replication**: high read/write ratio, but in combination with **simple queries**
- **Content-aware caching**: Check for queries at local database, and subscribe for invalidations at the server. Works good with **range queries and complex queries**.
- **Content-blind caching**: Simply cache the result of previous queries. Works great with **simple queries that address unique results** (e.g., no range queries).
- Question: What can be said about replication vs. performance?