Unit 3: Web and Service Application Architectures

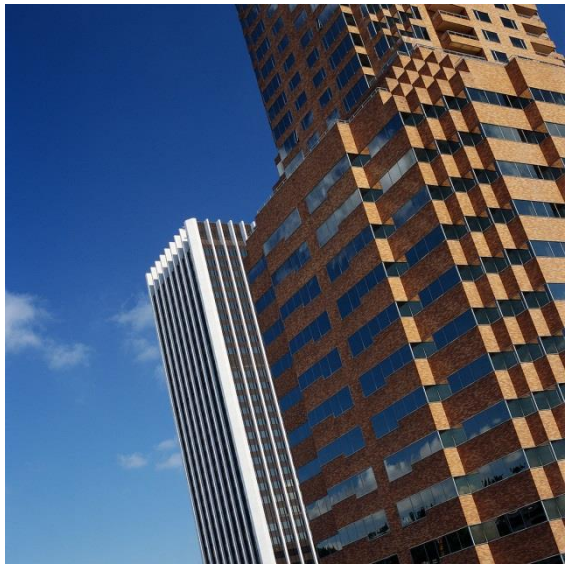
Introduction. Web Application Architectures. Service Application Architectures. Cloud Computing.

Unit 3: Web and Service Application Architectures

- Introduction
  - Software Architecture
  - Architectural Views
- Web Application Architectures
- Service Application Architectures
- Cloud Computing

Software Architecture

Software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both.



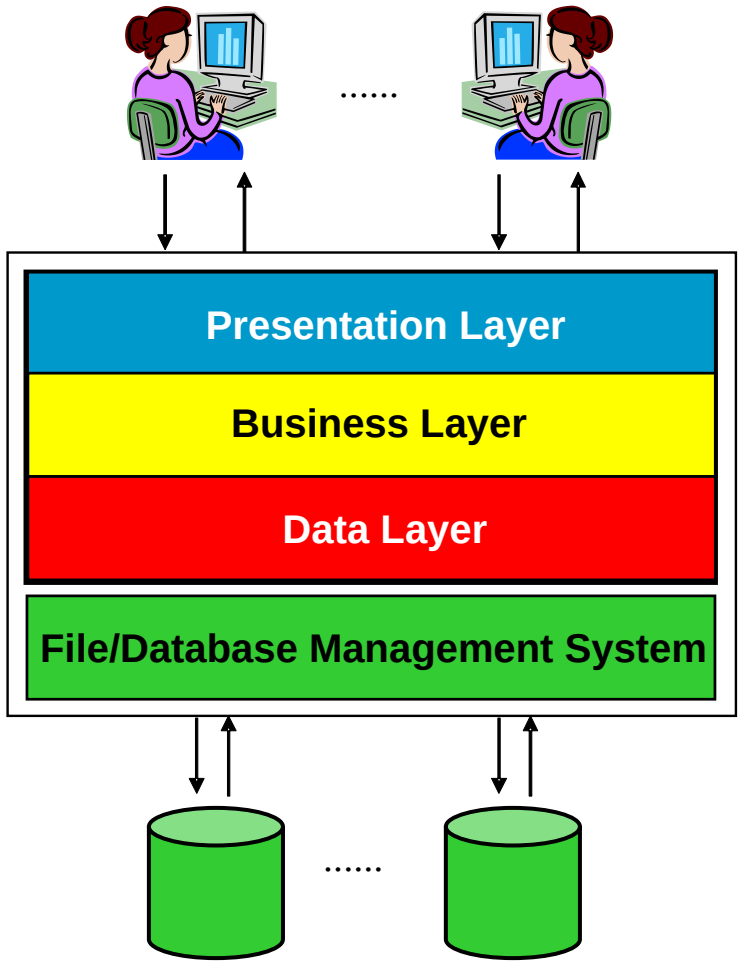
Architectural Viewpoints

- **Logical view** is the large-scale organization of the software components into packages, subsystems and layers that logically separate the functionality of a software system. It is called logical architecture because there is no decision about how these elements are deployed across different computational nodes.

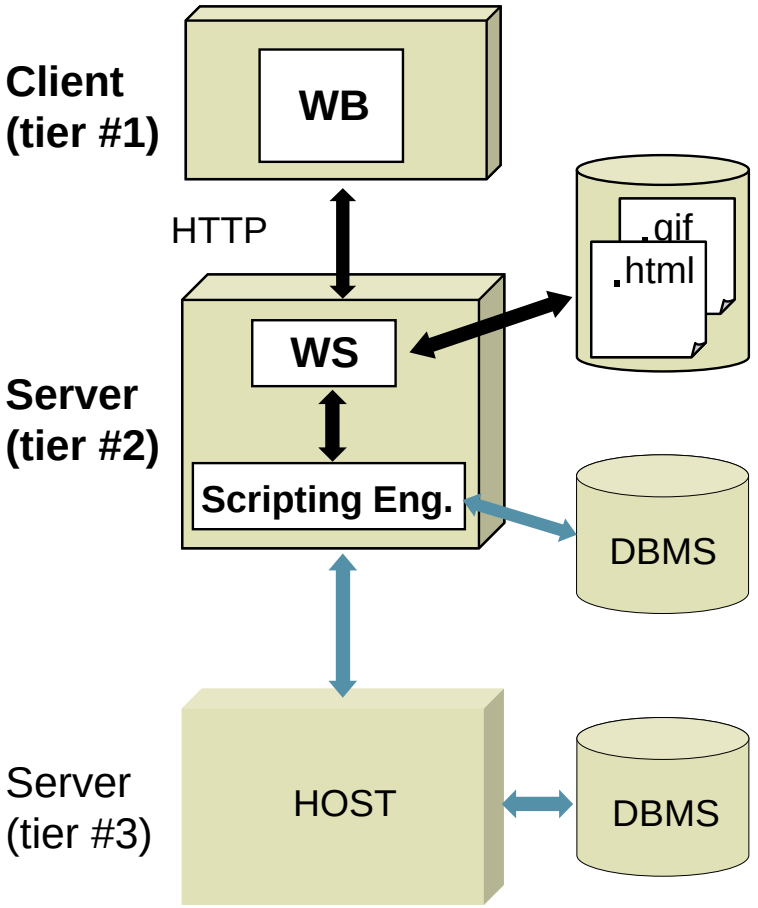
- **Physical view** is the organization and distribution of the logical architecture across different computational nodes in a network.

Architectural Viewpoints

3-Layered Logical Architecture



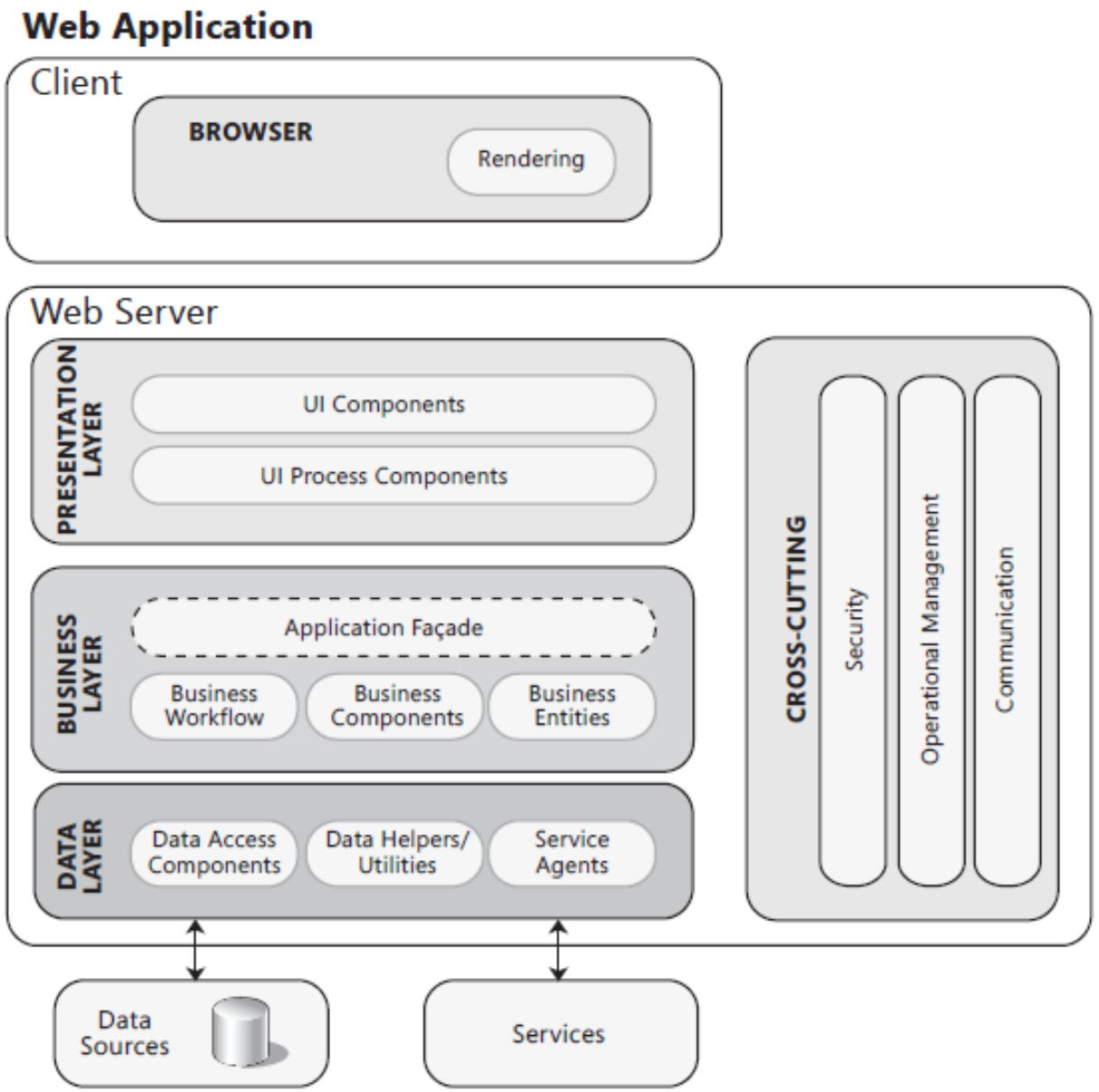
2(+)-Tiered Client/Server Physical Architecture



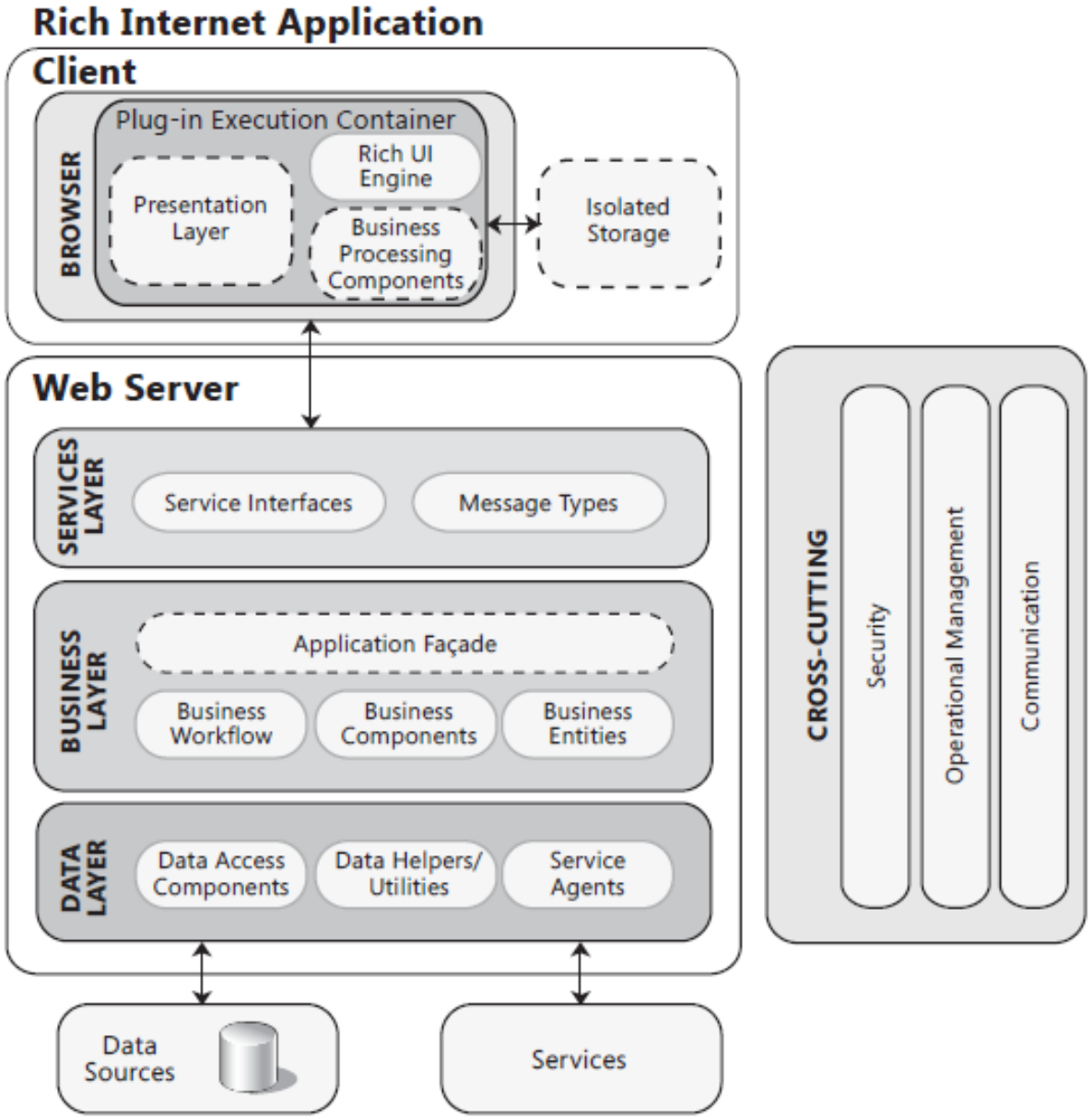
Unit 3: Web and Service Application Architectures

- Introduction
- Web Application Architectures
  - Logical - Physical Architecture
  - Dimensions to consider in a Physical Architecture Design
  - Architecture Patterns
    - Single server
    - Separate database
    - Replicated Web servers
    - Separated scripting engines
  - Web Caching
- Service Application Architectures
- Cloud Computing

Logical-Physical Architecture: “Classic” Web Apps



Logical-Physical Architecture: Rich Internet Apps



Dimensions in Physical Architecture Design

- **Non-functional requirements** that pursue the achievement of an adequate level of service
- Physical, financial and organizational **constraints** that may affect decision-making
- Alternative **scenarios** in architecture deployment

Non-functional requirements

- Performance: The app must sustain the expected workload defined in terms of
  - the maximum number of concurrent users,
  - the number of page requests served per unit of time,
  - the maximum time for delivering a page to the client
- Scalability: The architecture must be easily extensible
- Availability: Faults should not affect significantly the service delivered to users
- State maintenance (reliability): The state of the user interaction must be preserved, even when the application is distributed on multiple machines or failures occur
- Security:
  - Data should be protected
  - Users should be identified and granted access only to the data and functions they are entitled to

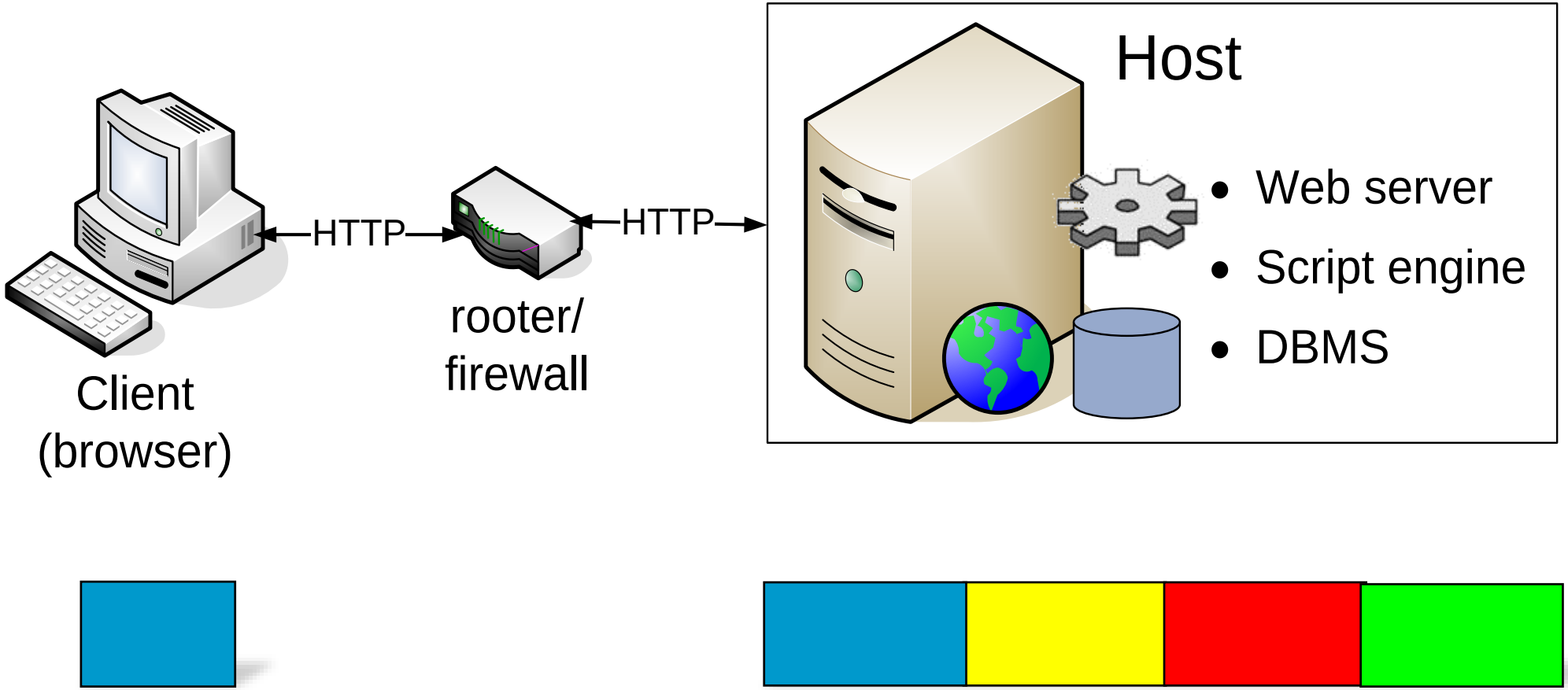
Constraints of Architecture Design

- Cost
  - Every configuration requires a different investment, in terms of processors, network infrastructure, interfaces, and software licenses
  - The application budget may limit the choice of hardware resources and software products
- Complexity
  - Some configurations are simpler than others to set up and maintain
  - The unavailability or the cost of specialized technical skills may constrain the architecture design
- Corporate standards and infrastructures
  - The WebApp may be deployed within a corporate IT infrastructure, which may constrain the selection of hardware resources and software products.

Scenarios of Architecture Deployment

- Internal
  - The application architecture is kept inside the enterprise and maintained by the internal IT department.
- Housed
  - The application architecture is maintained by the internal IT department of the enterprise, but is physically installed at an external service provider.
- Hosted
  - The application architecture is located at the premises of an external service provider, who also maintains it.

Architecture Pattern: Single Server



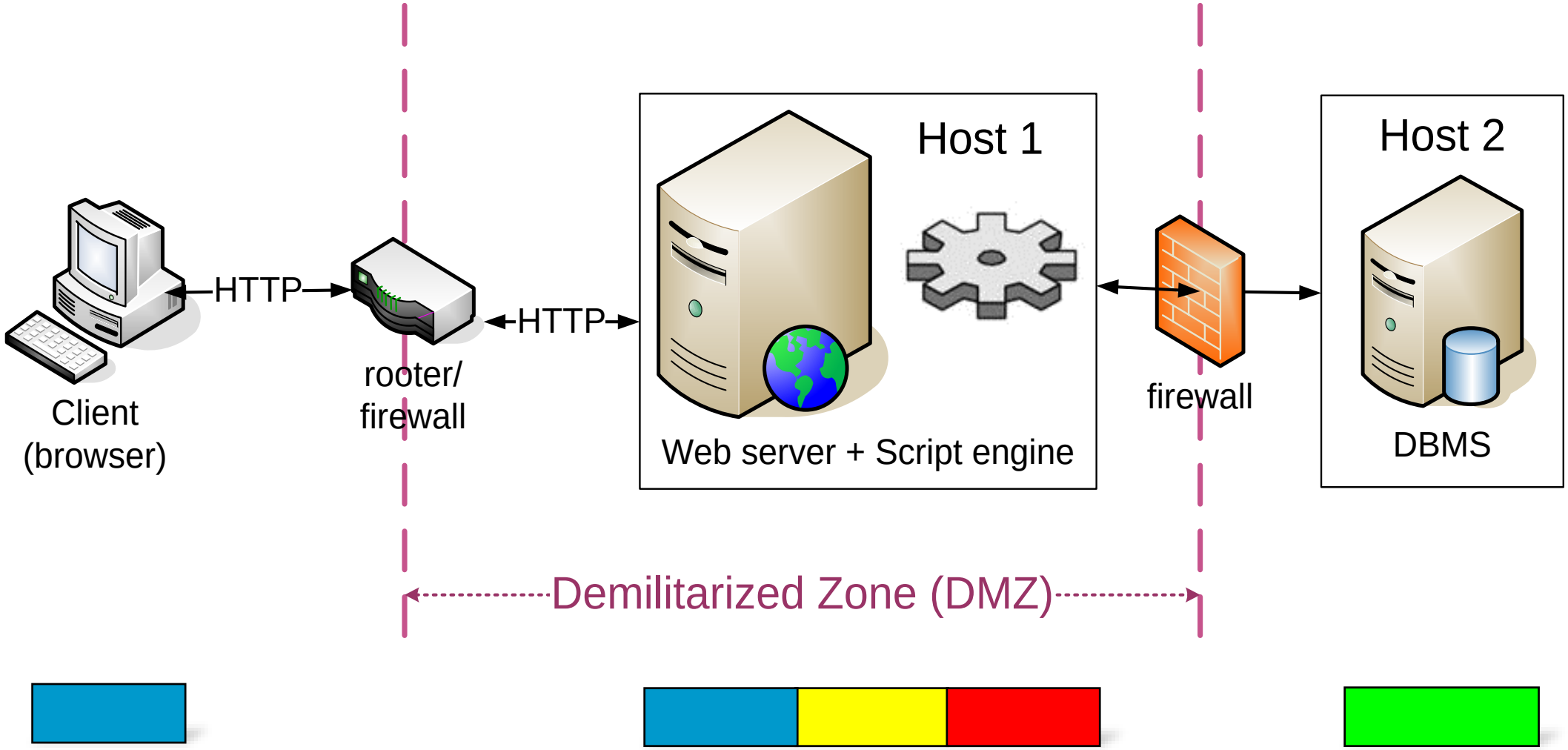
Single Server Pattern: Evaluation (1/2)

- Performance
  - Depends on the configuration of the server: CPU speed, available memory, disk access latency, etc.
  - The DBMS is both memory and CPU-intensive
- Scalability
  - Is bound by the hardware architecture of the selected server
- Availability
  - Every software and hardware element is a single point of failure: if it breaks, the entire system hangs.
  - Can be improved by adding redundant hardware resources (multiple CPUs, mirrored disks) and by installing multiple processes running different instances of the Web server, script engine, and database …

Single Server Pattern: Evaluation (2/2)

- State maintenance
  - No problems with none of the three possibilities to store user data: client, server or database
- Security
  - A weak aspect of this configuration: attackers breaking the firewall and the Web server can take control of the host and gain direct access to the database, violating data protection
- Low cost, as far as massive parallelism is not required.
- Low complexity

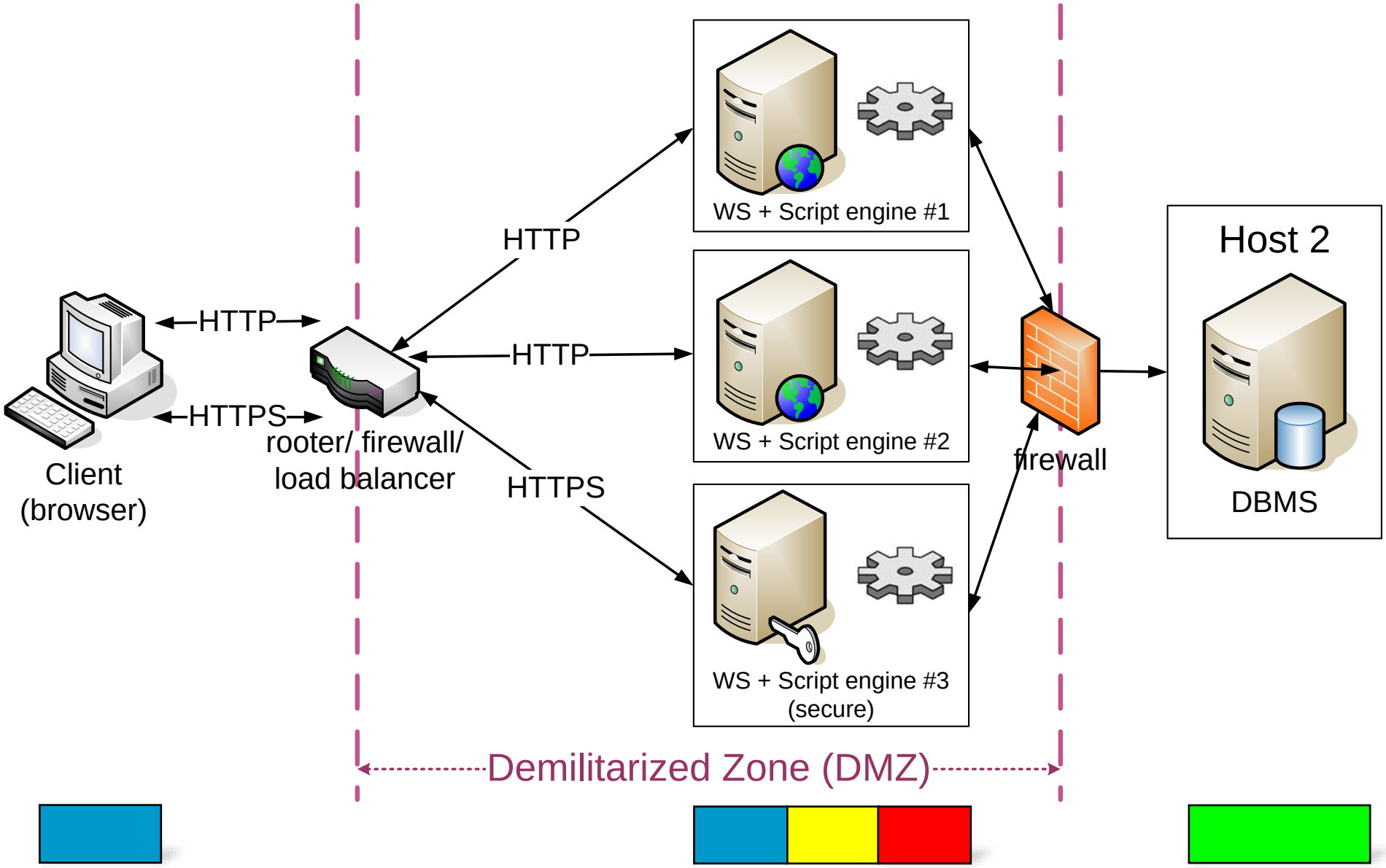
Architecture Pattern: Separate Database



Separate Database Pattern: Evaluation

- Better performance
  - One extra machine
  - Each tier can be tuned to the requirements of the installed software
- More scalable
  - It is possible to act separately on each tier.
  - Normally, the first bottleneck is in the middle tier
- Availability is not improved
  - Each component is still a single point of failure
- Significantly improved security
  - The inner firewall may disallow HTTP requests at all and let only database requests pass, making it more difficult for attackers to reach the data tier.

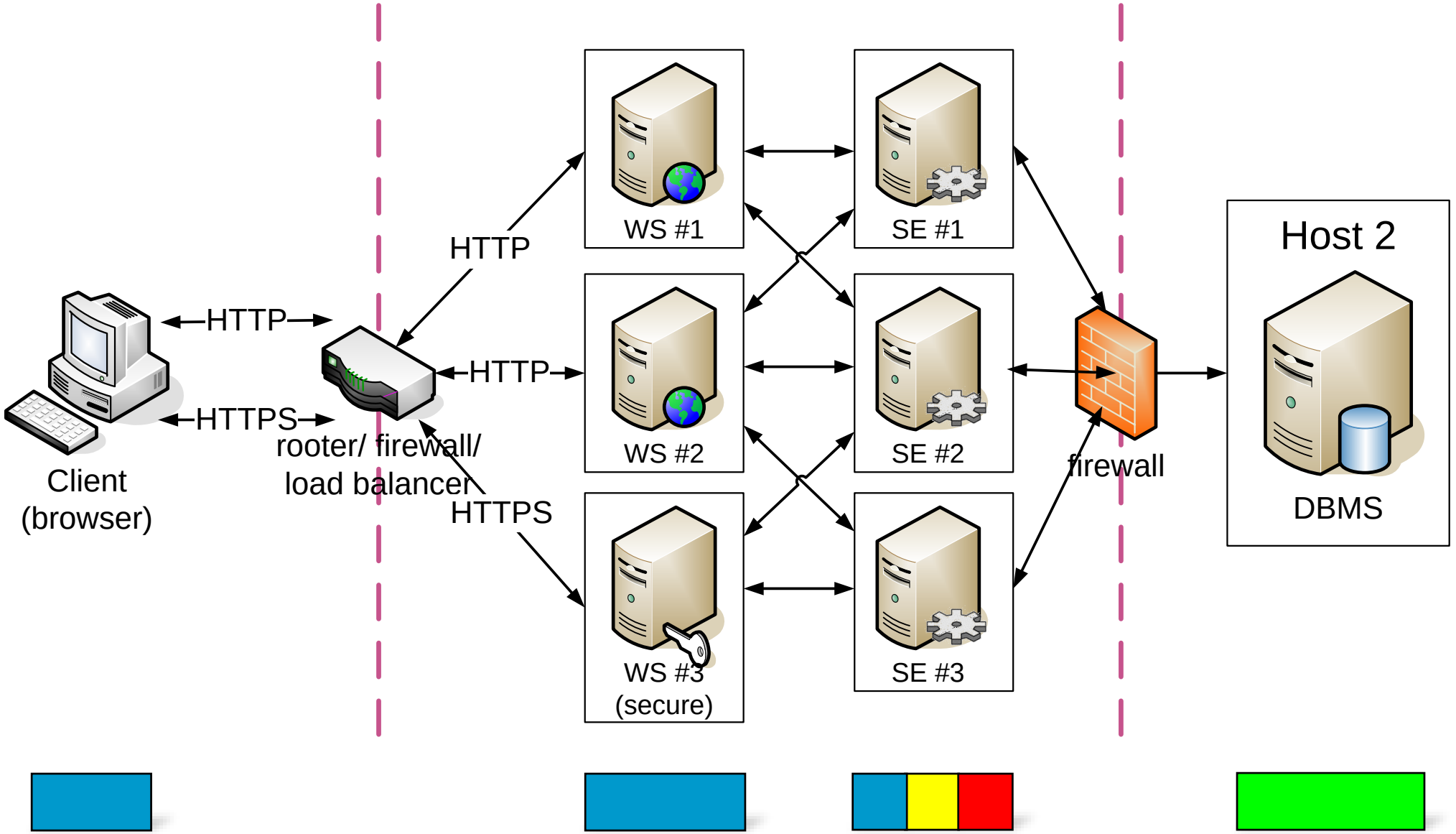
Architecture Pattern: Replicated Web Server



Replicated Web Server Pattern: Evaluation

- Improved performance and scalability:
  - Load balancing
  - Clustering: A cluster is a group of servers (aka nodes) that provide a unified view of the services that they individually offer
- Improved availability:
  - Fail-over: if a cluster node fails, its workload can be redistributed to the other nodes of the same cluster
- Session state maintenance on the replicated servers
  - Session affinity (aka sticky sessions): The load balancer sends all the incoming requests pertinent to a given session to the same server.
  - Session migration: Session state is shared by the servers in the cluster
- Improved data transmission security
  - One of the Web servers may be configured to handle the connections that require cryptographic protection.

Architecture Pattern: Separate Script Engine



Separate Script Engine Pattern : Evaluation

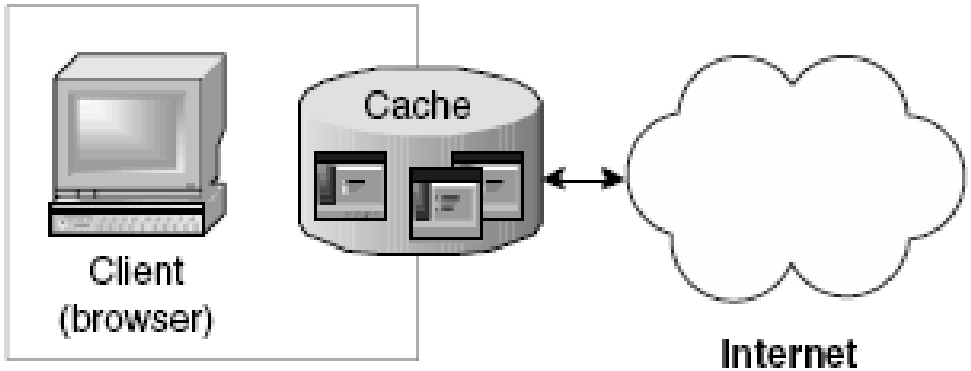
- Improved performance, scalability and availability
  - Web server and the scripting engine can be replicated independently so that the number and configuration of the hosts can be optimized: a well-balanced configuration may require more machines for the scripting engines than for the Web servers.
  - The communication overhead introduced by the separation should be compensated by the performance increase.

Web Caching

- Caching consists of temporarily storing resources in a fast access location, for later retrieval.
- Benefits:
  - Reduction of the response time
  - Reduction of computation effort when the resource is dynamically build
- Many things can be cached:
  - Static HTML pages and multimedia files.
  - Fragments of pages computed by scripting programs.
  - Intermediate data consumed by the scripting programs for producing page, e.g. XML files.
  - The result of database queries or other application commands

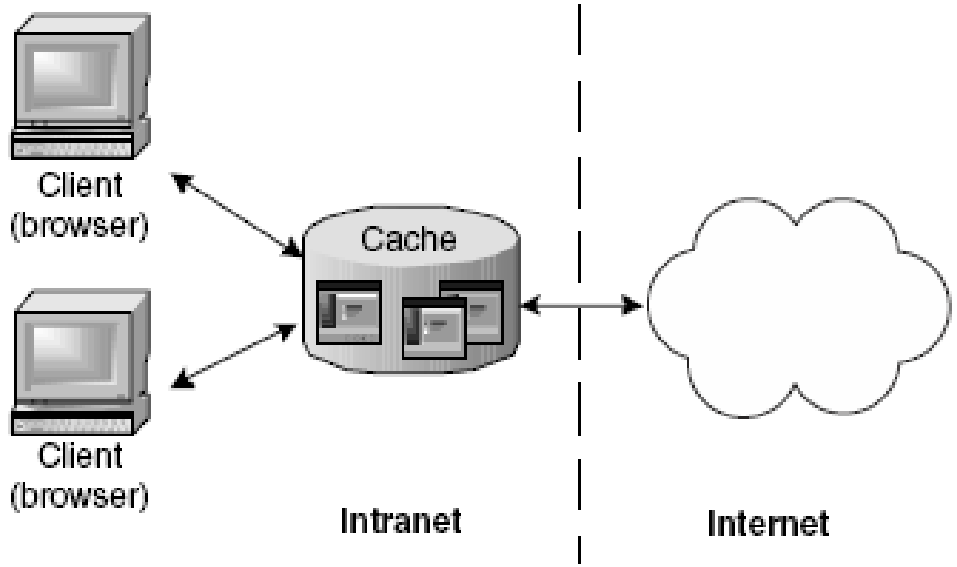
Web caching: Where to Cache (1/3)

Browser caching



Every Web browser contains a cache of HTML pages and multimedia files used to speed up the rendition of pages that contain cached objects.

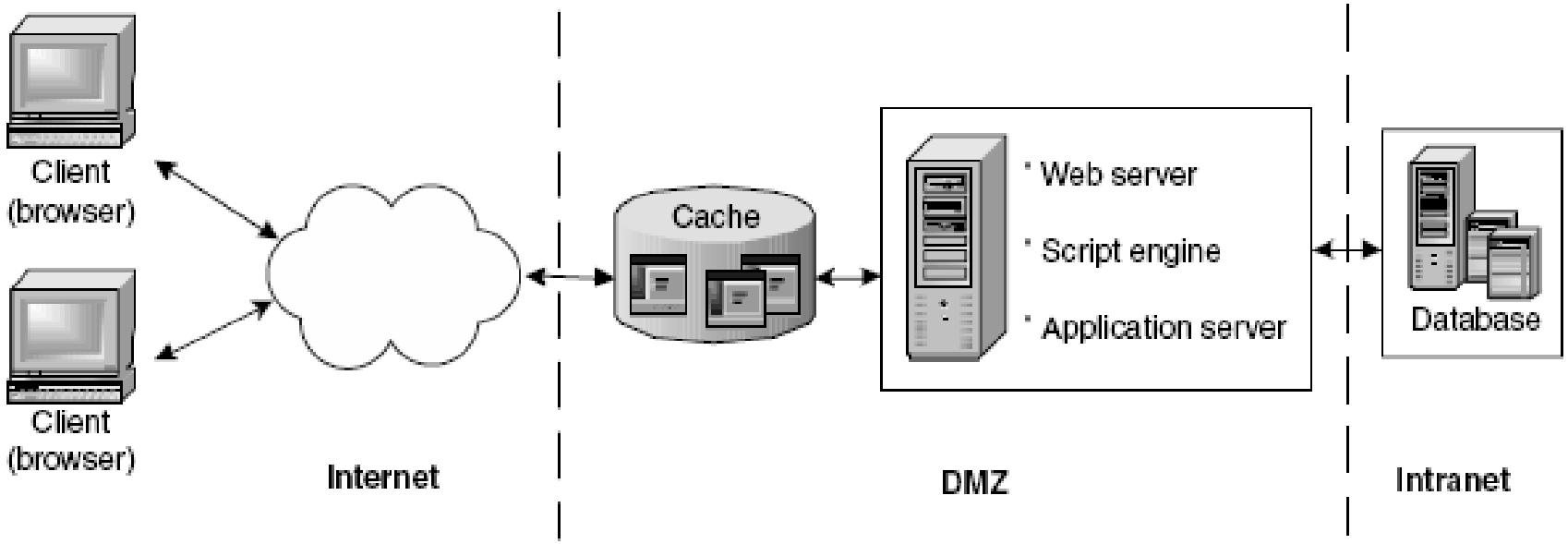
Proxy caching



Proxy caches store a local copy of each resource requested by users, and avoid accessing the Internet for retrieving frequently asked pages

Web caching: Where to Cache (2/3)

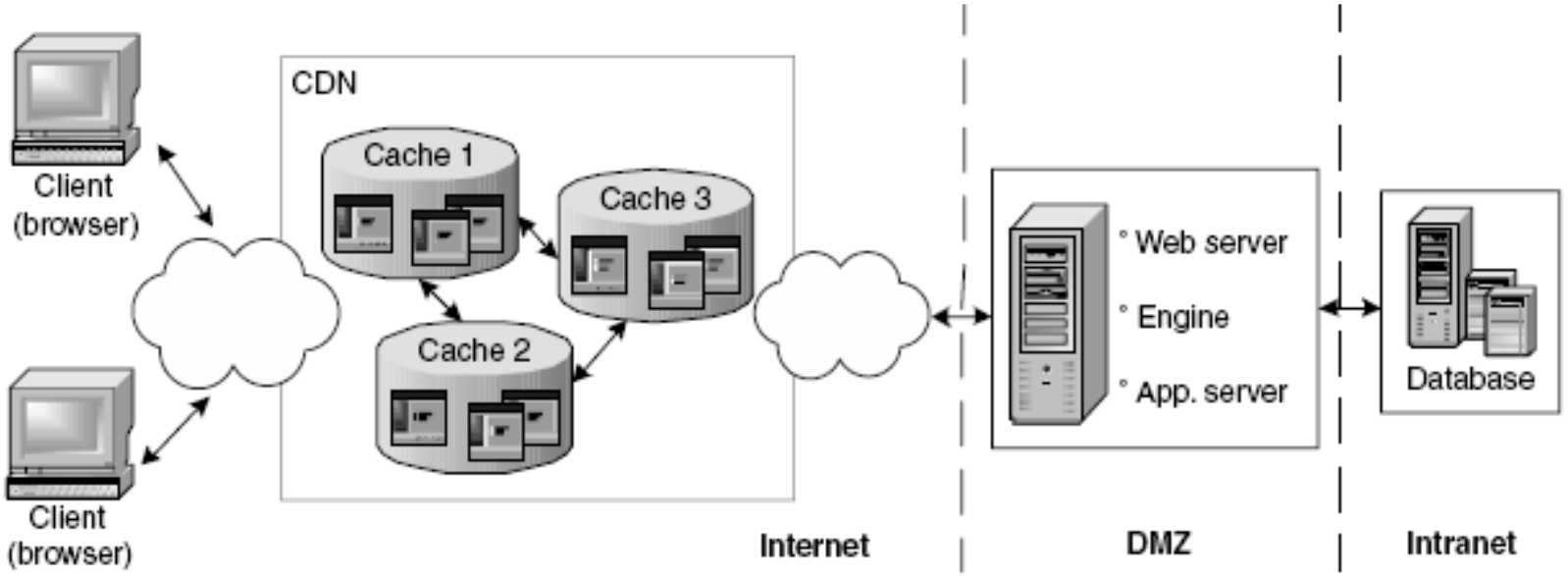
Server accelerators



- A server accelerator is a “buffer” placed in front of a server cluster that intercepts requests, caches copies of the objects produced by the servers, and delivers them to the subsequent requests.
- Page prefetching: Based on the last request, the server accelerator loads into cache those pages that are more probable of being requested next.

Web caching: Where to Cache (3/3)

Content delivery networks (CDNs)

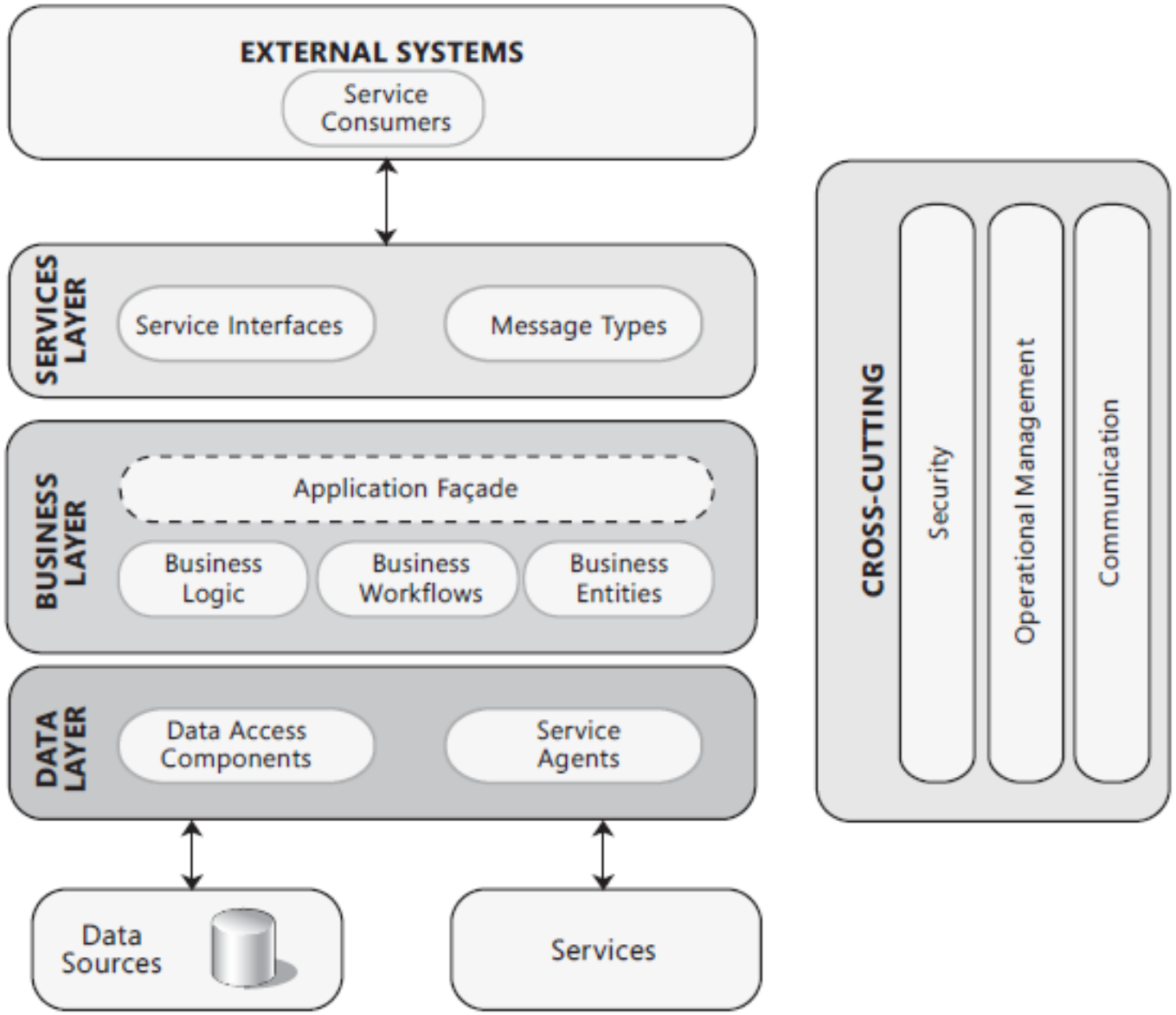


- CDNs are systems of computers networked together across the Internet that allow content providers to outsource their caching infrastructures
- When a client requests a page to the origin server, this returns a page with rewritten links that point to the nodes of the CDN
- The CDN serves requests selecting the optimal copy of the page by taking into account the geographical location of the user and the real-time traffic conditions.

Unit 3: Web and Service Application Architectures

- Introduction
- Web Application Architectures
- Service Application Architectures
  - Logical - Physical Architecture
  - Dimensions to consider in a Physical Architecture Design
  - Architecture Patterns
- Cloud Computing

Logical-Physical Architecture: Service Apps



Dimension in Physical Architecture Design

- **Non-functional requirements**, **constraints** and alternative **scenarios** in architecture deployment are the same as for Web applications

Architecture Patterns

- Service applications may be deployed using the same architectural patterns as for Web applications considering the performance and security issues inherent in distributed scenarios
- It is highly recommended to use those architectural patterns that deploy the service layer on the same tier as the business layer. It improves application performance

Unit 3: Web and Service Application Architectures

- Introduction
- Web Application Architectures
- Service Application Architectures
- Cloud Computing

Cloud Computing – A Definition

"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models"

The NIST Definition of Cloud Computing

Cloud Computing – 5 Essential Characteristics

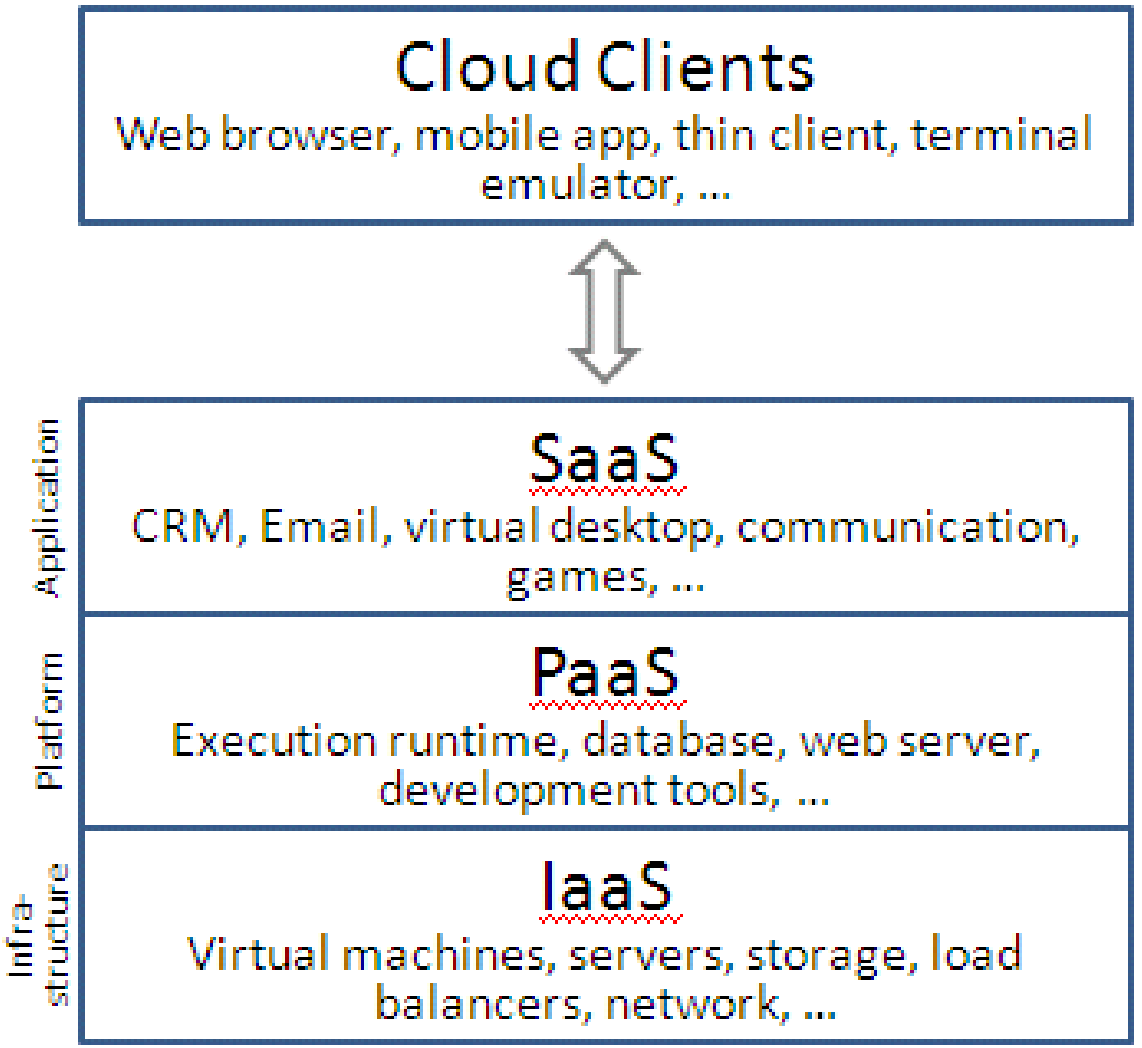
- **On-demand self-service**: A consumer can unilaterally provision computing capabilities as needed, such as server time and network storage, as needed automatically without requiring human interaction
- **Broad network access**: Capabilities are available over the network and accessed through standard mechanisms.
- **Resource pooling**: Physical and virtual resources are dynamically assigned and reassigned according to consumer demand.
- **Rapid elasticity**: Capabilities can be rapidly and elastically provisioned.
- **Measured Service**: Resource usage is monitored, controlled, and reported providing transparency for both the provider and consumer of the utilized service.

Cloud Computing – Service Models

Software as a Service

Platform as a Service

Infrastructure as a Service



http://www.wikipedia.org/

Cloud Computing – Deployment Models

- **Private cloud**: provisioned for exclusive use by a single organization
- **Community cloud**: provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns
- **Public cloud**: provisioned for open use by the general public.
- **Hybrid cloud**: composition of two or more distinct cloud infrastructures (private, community, or public) bound together by standardized or proprietary technology that enables data and application portability

References

- **Microsoft Application Architecture Guide, 2nd Edition** http://msdn.microsoft.com/en-us/library/dd673617.aspx
- S. Ceri et al. **Designing Data-Intensive Web Applications**. Capítol 10. Morgan Kaufmann, 2003.
- National Insitute of Standards and Technology. **Cloud Computing Definition** http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf