Client-Server Architecture

Lotfi ben Othmane

Learning Outcomes

1. Understand client/server architecture

What is Software Architecture?

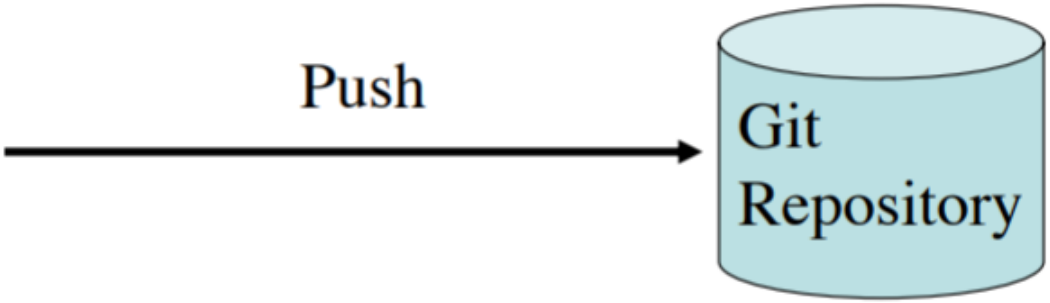
Architecture is defined by the recommended practice as the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution.

ANSI/IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems

Reference Architecture

Reference architecture provides an overall logical structure for a particular type of application.

Single-Process Architecture

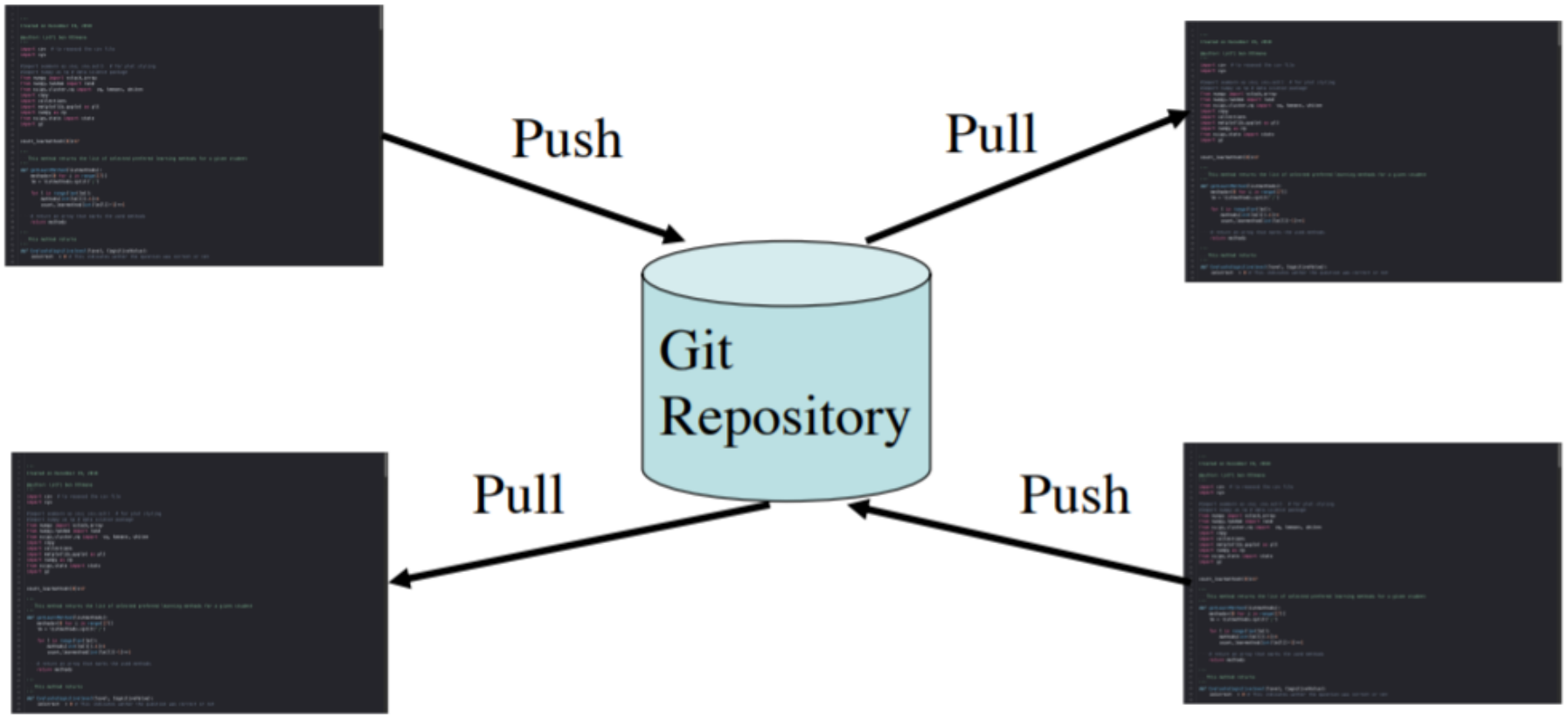


Single-Process Applications

Reference architectures for single-process applications:

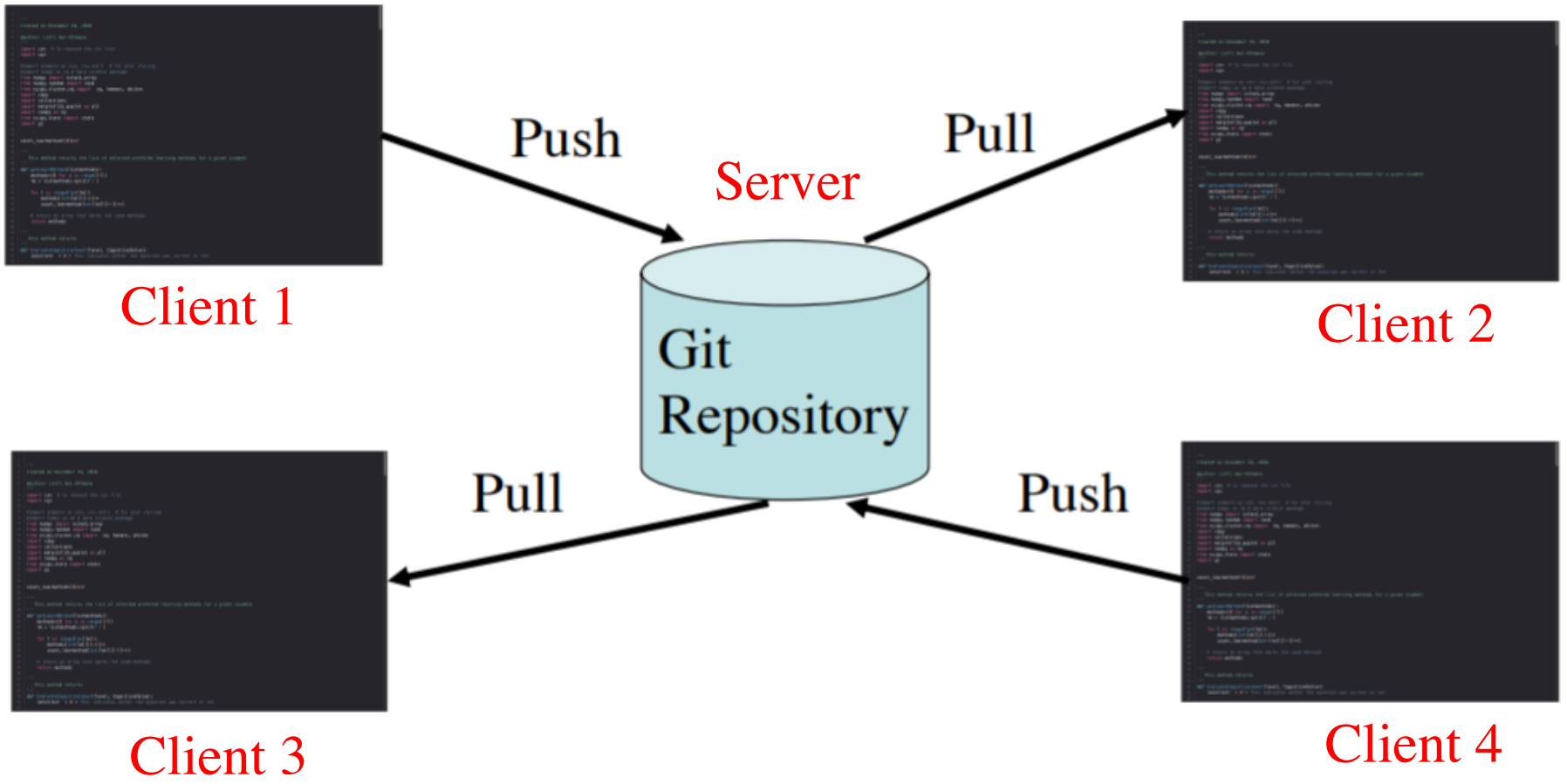
1. Desktop application
2. Embedded system
3. Mobile application
4. Service

Client/Server Application



Is this git a desktop application?

Client/Server Application



One server and a set of clients

Client-Server Style

- A client-server style is a style that involves client processes and a server process that communicate through a network. A server could be accessed by many clients and a client can access many servers.

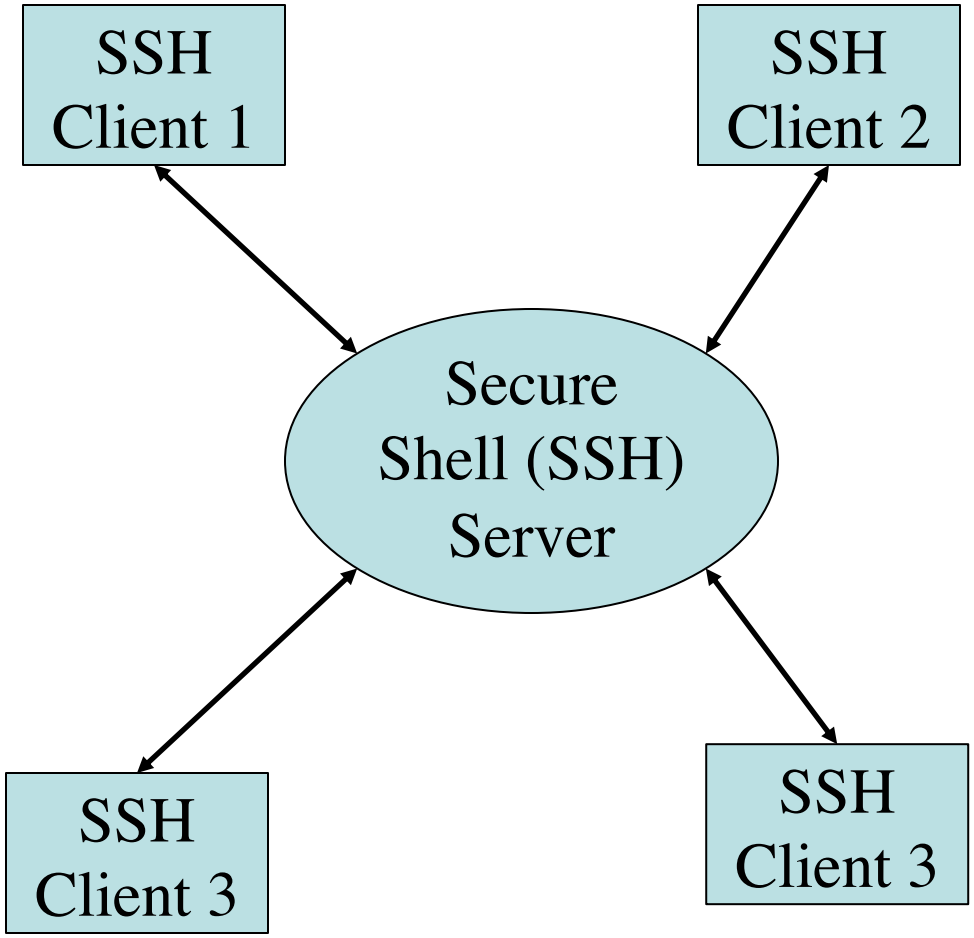
- A client sends requests, gets replies, and processes them.

- Examples: Email, database systems, etc.

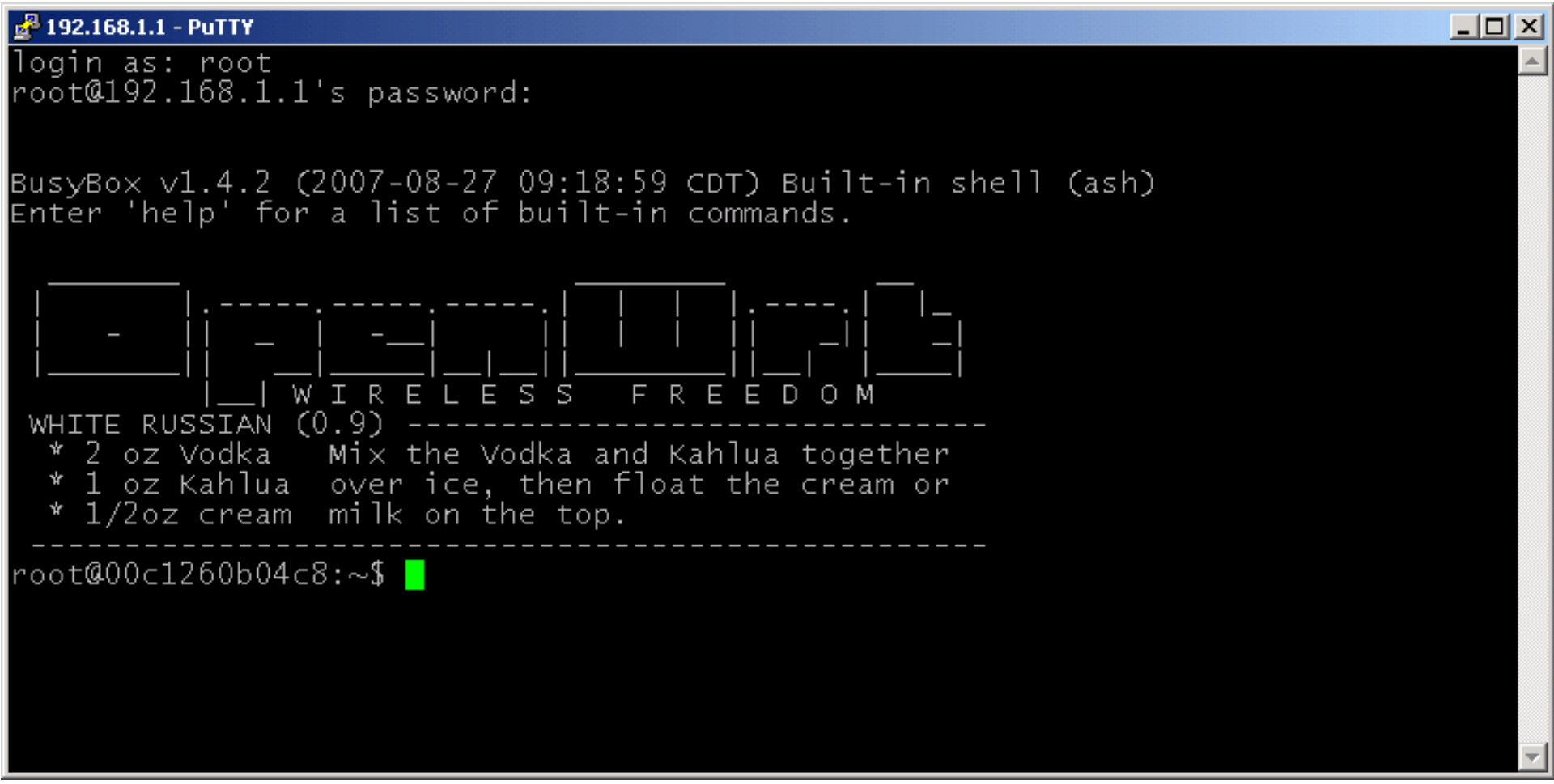
- Variation of structures: Application servers, P2P (peer to peer), Client-Queue-Client-System (e.g. chat).

Client-Server Architecture

Client-server model is a distributed application model where tasks are partitioned between the client and server.



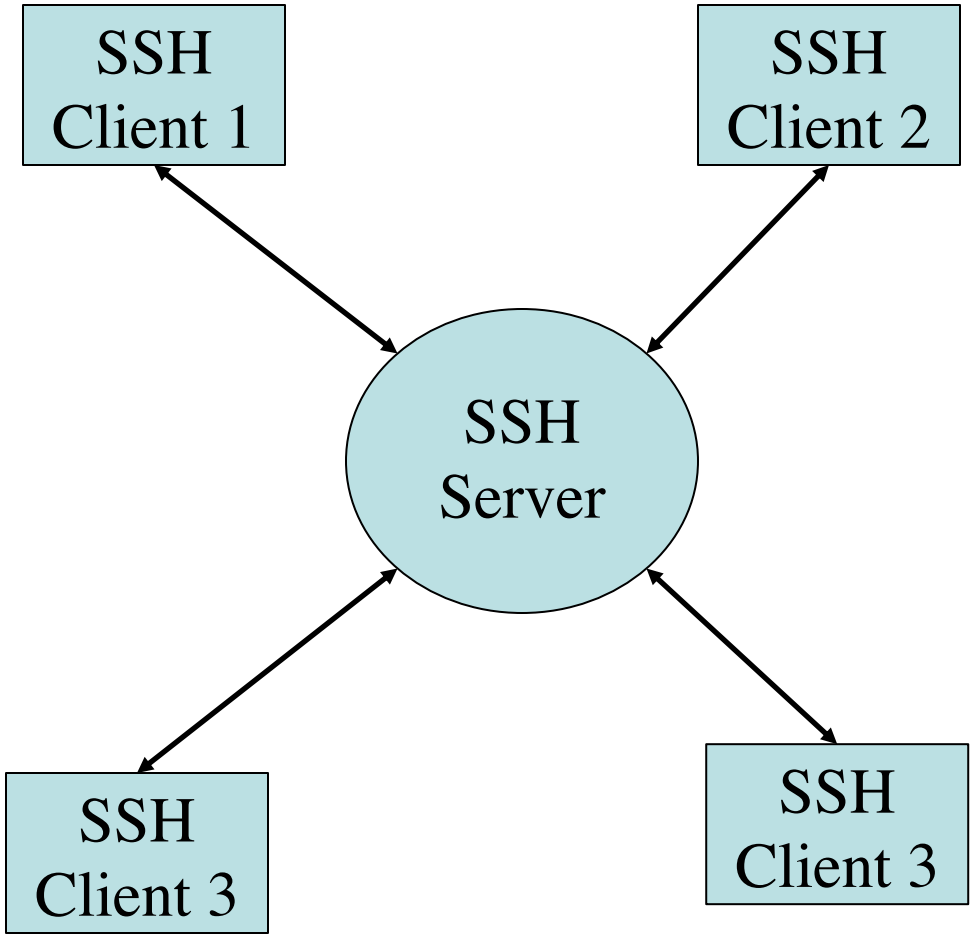
Putty – SSH Client



https://en.wikipedia.org/wiki/Secure_Shell

Client-Server Architecture

What operations does ssh need to support?



Client-Server Architecture

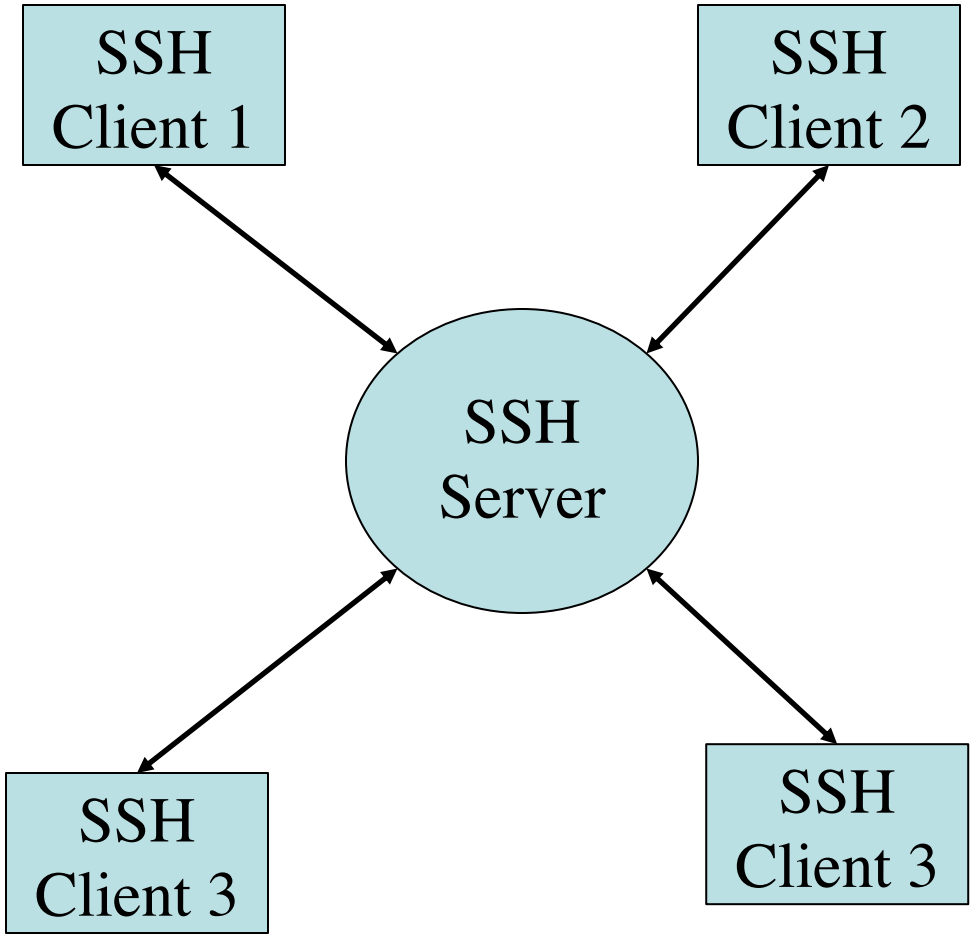
SSH Operations

- Login
- Send commands
- Display command results
- Logout
- Create application session
- Execute commands
- Close session

Example of operations to be performed by the client and server.
Note: There is no order of the lists' items.

Exercise – Client-Server Architecture

Now we need to split the operations between the client component and server component. Then we need to specify the interactions between both components



Client-Server Architecture

Client

- Send username
- Send password
- Send commands
- Set application session
- Display command results
- Request logout

Server

- Accept username
- Verify password
- Create application session
- Execute commands
- Close session

Example of operations to be performed by the client and server.
Note: There is no order of the lists' items.

Uses of the Style

You should consider the style to:

- Support many clients (web applications, business processes…).
- Centralize data and management functions.
- Support many different types of clients.

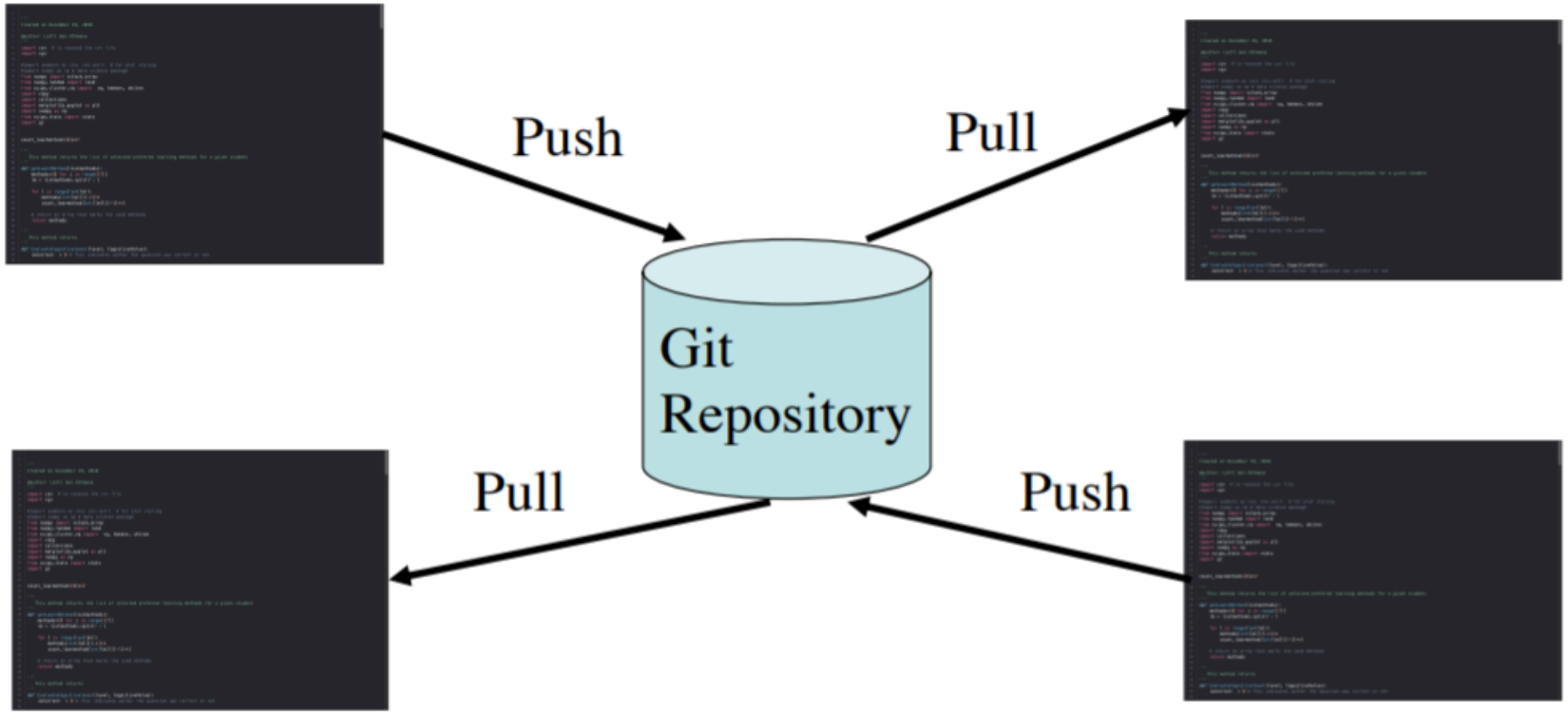
Uses of the Style – Cont.

Problems associated with sharing resources include:

- Managing the availability of resources for the clients at the server
- Managing the concurrency to access the resources
- Controlling the performance of the server
- Managing accesses to the clients scopes
- Diversity of the clients operating systems
- And more…

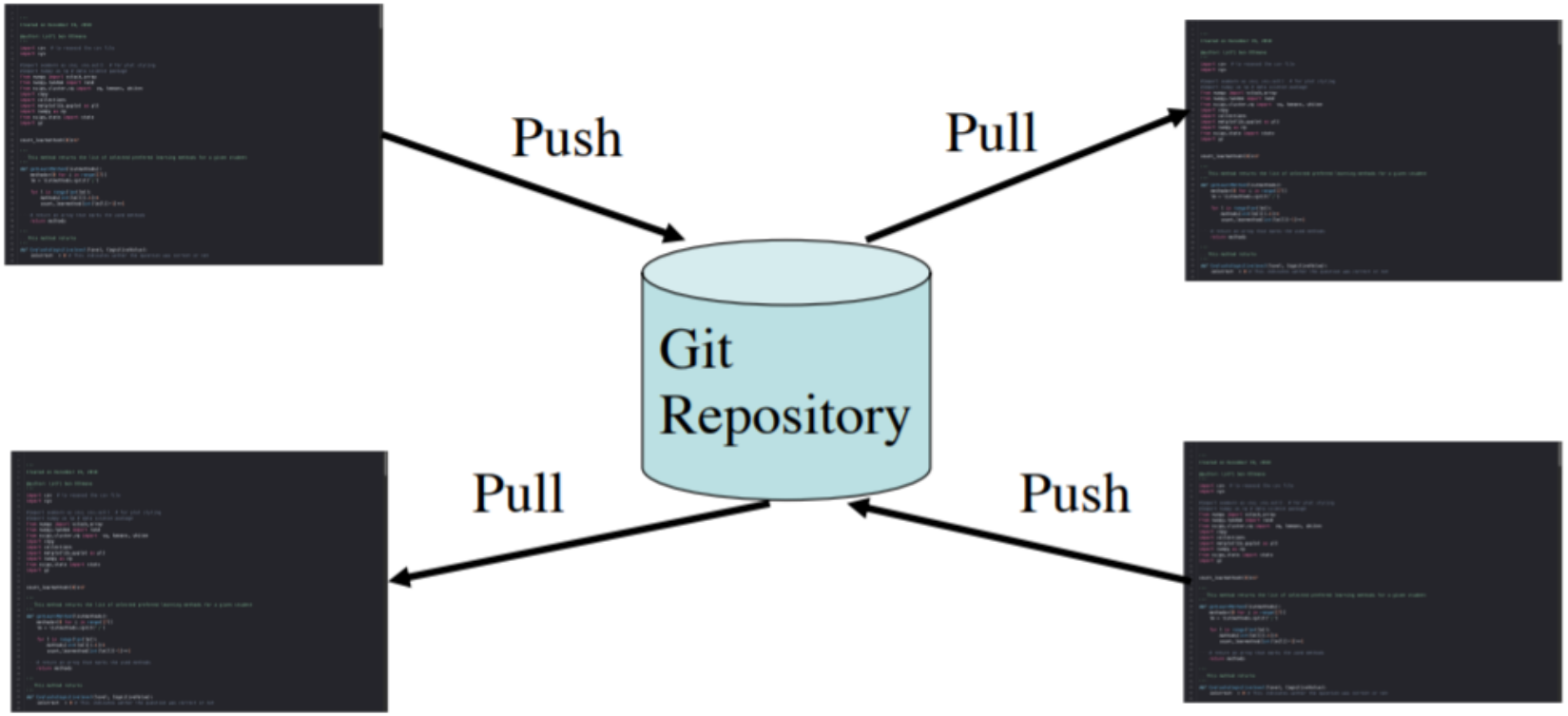
How would you solve these problems?

Benefits - Ease of Data Management



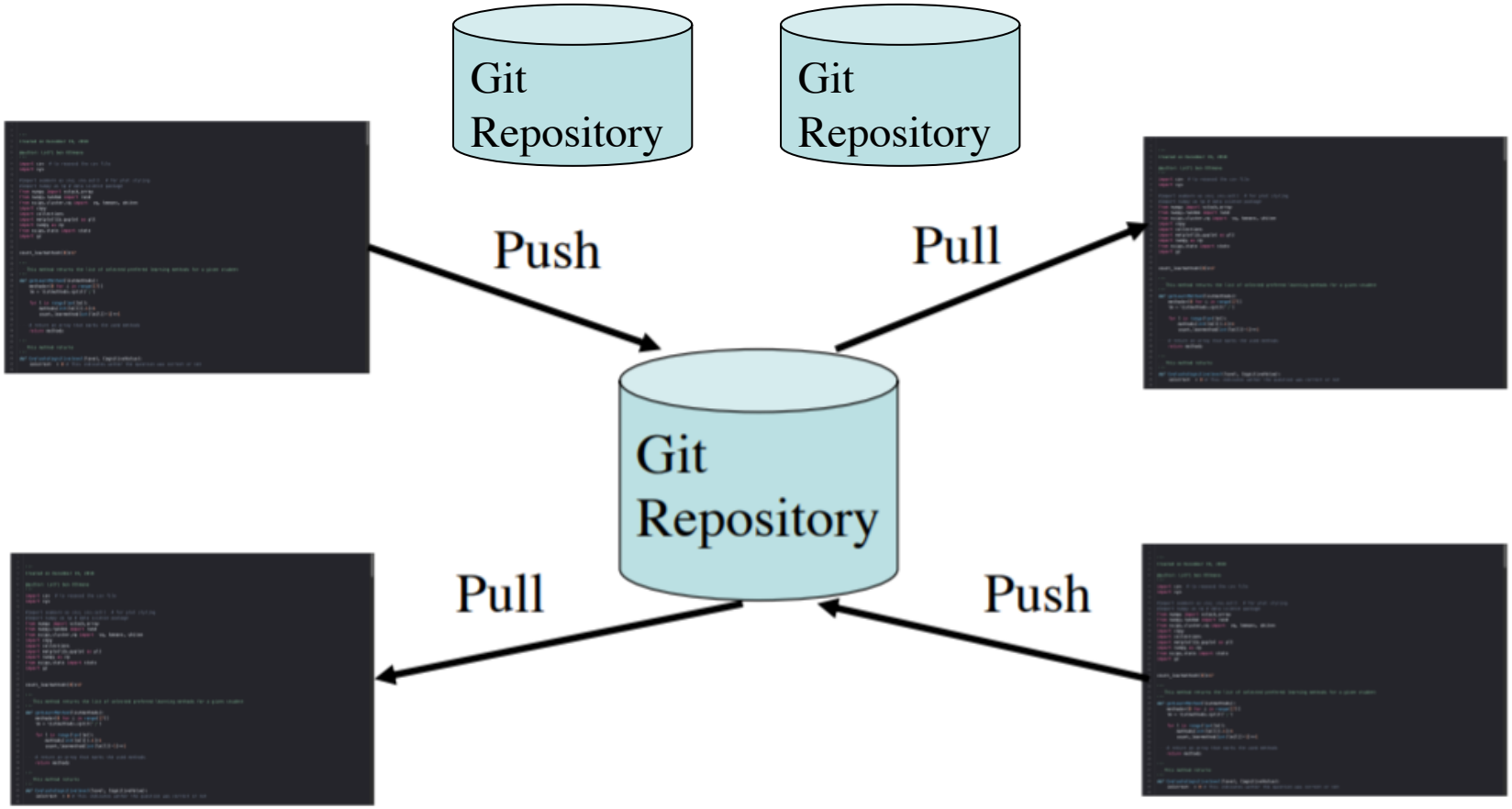
One process can manage several accesses to the data

Benefits - Ease of Maintenance



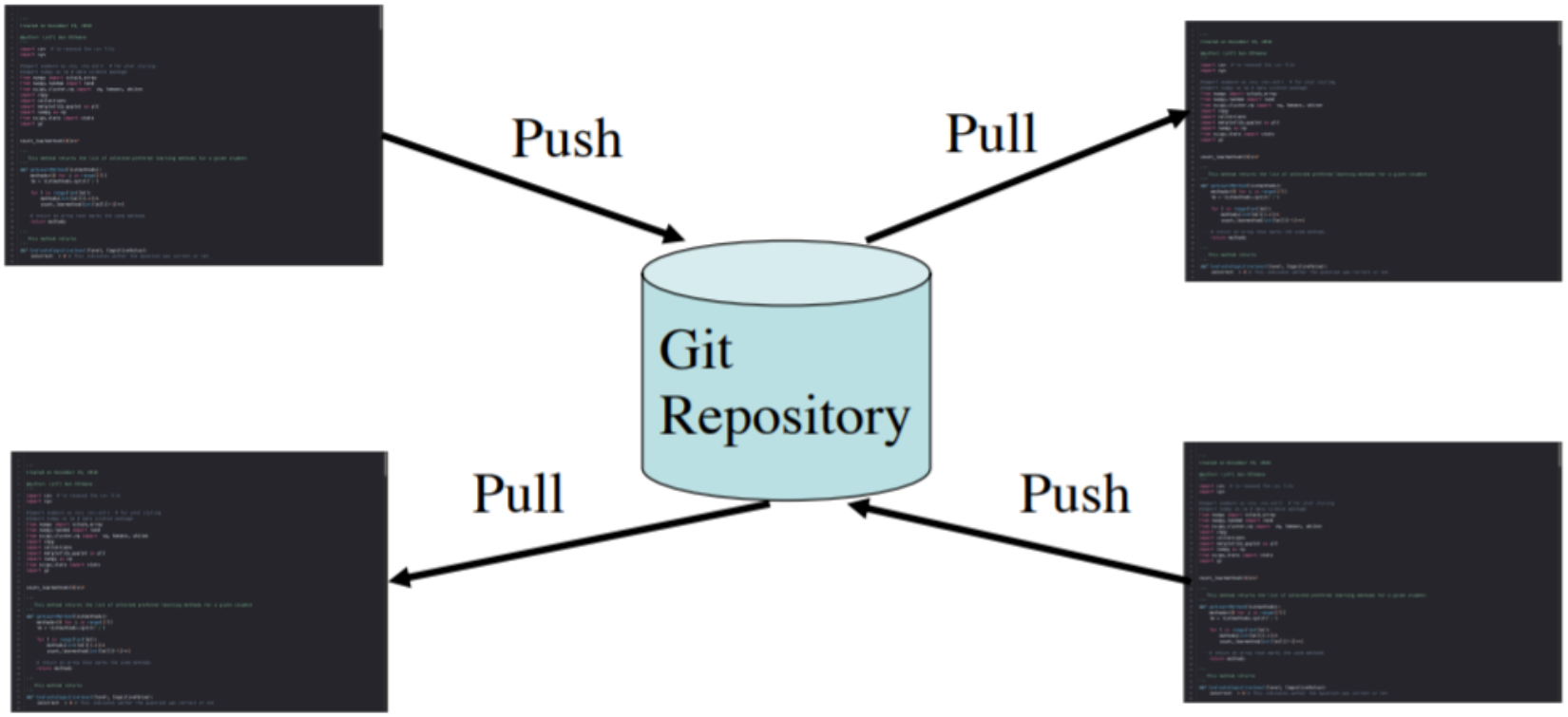
We deploy only one application server.
Each user can install a version of the client.

Disadvantages - Scalability



Scaling the server component requires scaling both the data and the business logic together.

Disadvantages - Reliability



Failure of THE server affects all the users.

History

We wanted to share documents but,

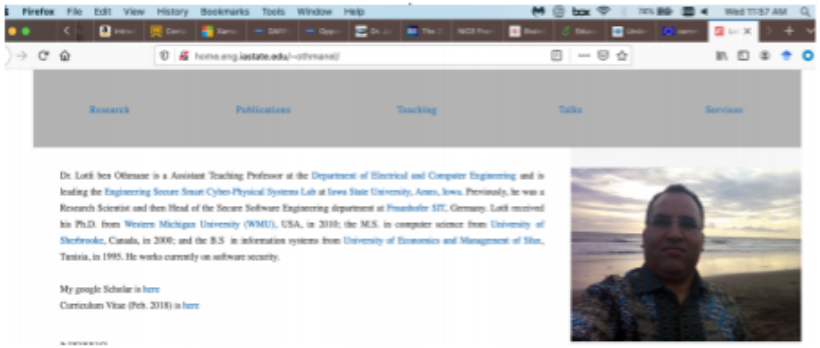
- Do not have physical access to clients.
- Do not want to deploy the client application on the desktops.
- Need to support different hardware types.
- Need to support different operation systems (Linux, OS2, DOS/Windows, MAC OS).

Simple Application – Web Sites

Web browsers communicate with a server using the HTTP protocol

Server

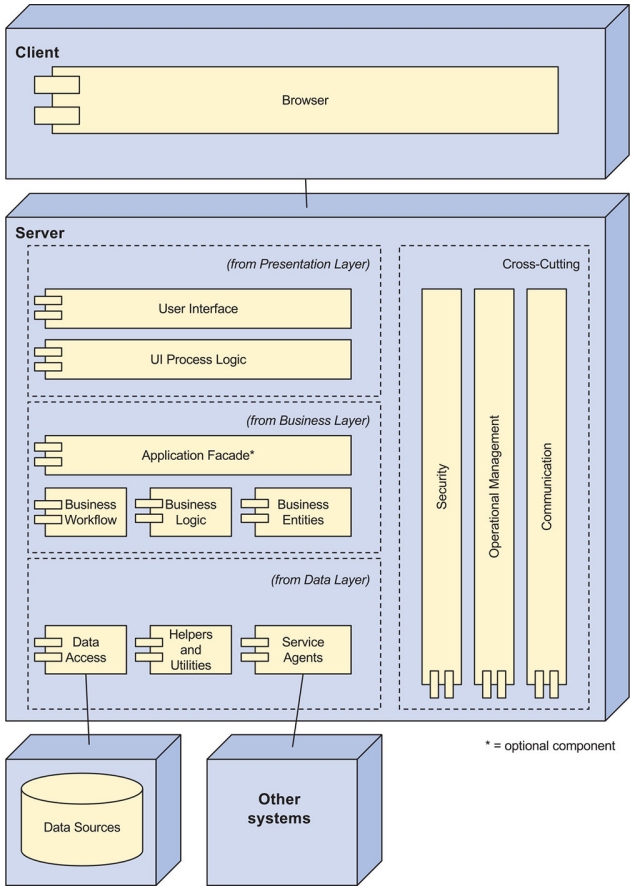
HTTP + HTML



Web Browser

Web Applications

- We want to get data from the user and respond based on their requests – dynamic web pages.
- Each component is assigned a set of responsibilities.



Web Applications

Use web applications when:

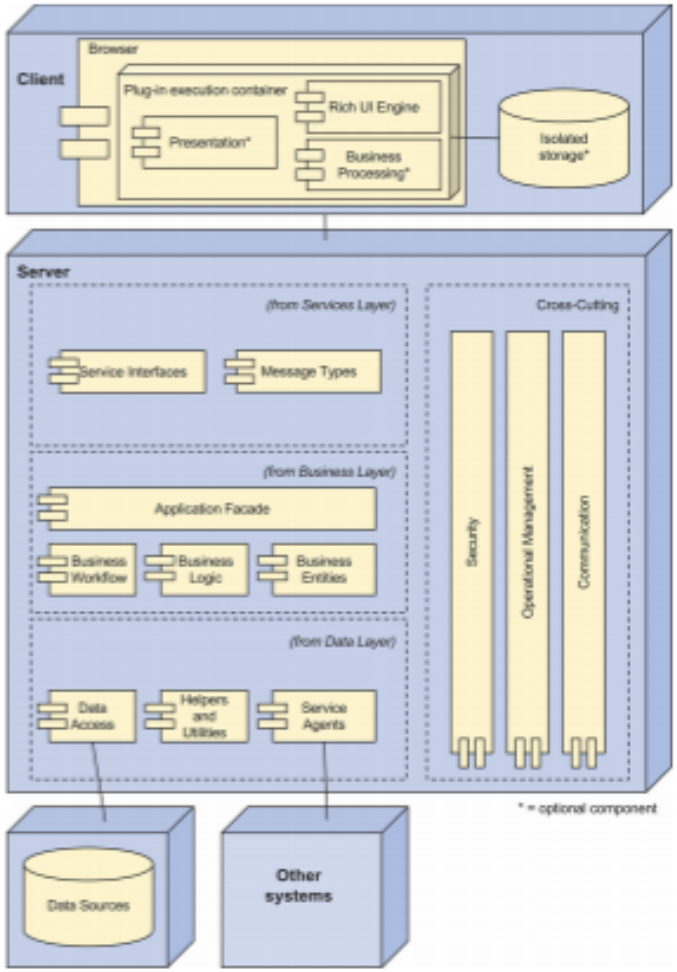
1. You do not want to deploy software at the client host
2. There's a need for the user interface to be portable
3. The application can be used over the Internet
4. Its ok to have **restricted** access to resources at the client

But what if:

1. ~~You do not want to deploy software at the client host~~
2. There's a need for the user interface to be portable
3. The application can be used over the Internet
4. ~~Its ok to have **restricted** access to resources at the client~~
5. There is a need to access resources of the client

Rich Client Applications

- Rich internet applications typically run inside a browser. They support rich user interaction and business logic.
- Some business logic may be executed on the host machine.
- May use local resources.



Rich Client Applications

Reasons to use rich client applications:

- We need a rich interface that runs in a browser.
- We need to perform complex business logic on the client's machine.
- The deployment of the application is simple.
- Loading time is acceptable.

So…

Architecture references (like client/server, web app) solve common challenges through the structure of the software: sharing data and processing, portability and deployability.

Architects fit their components into the reference architecture and use existing technologies such as web servers and database management systems as third-party software to address their needs.

Thank You