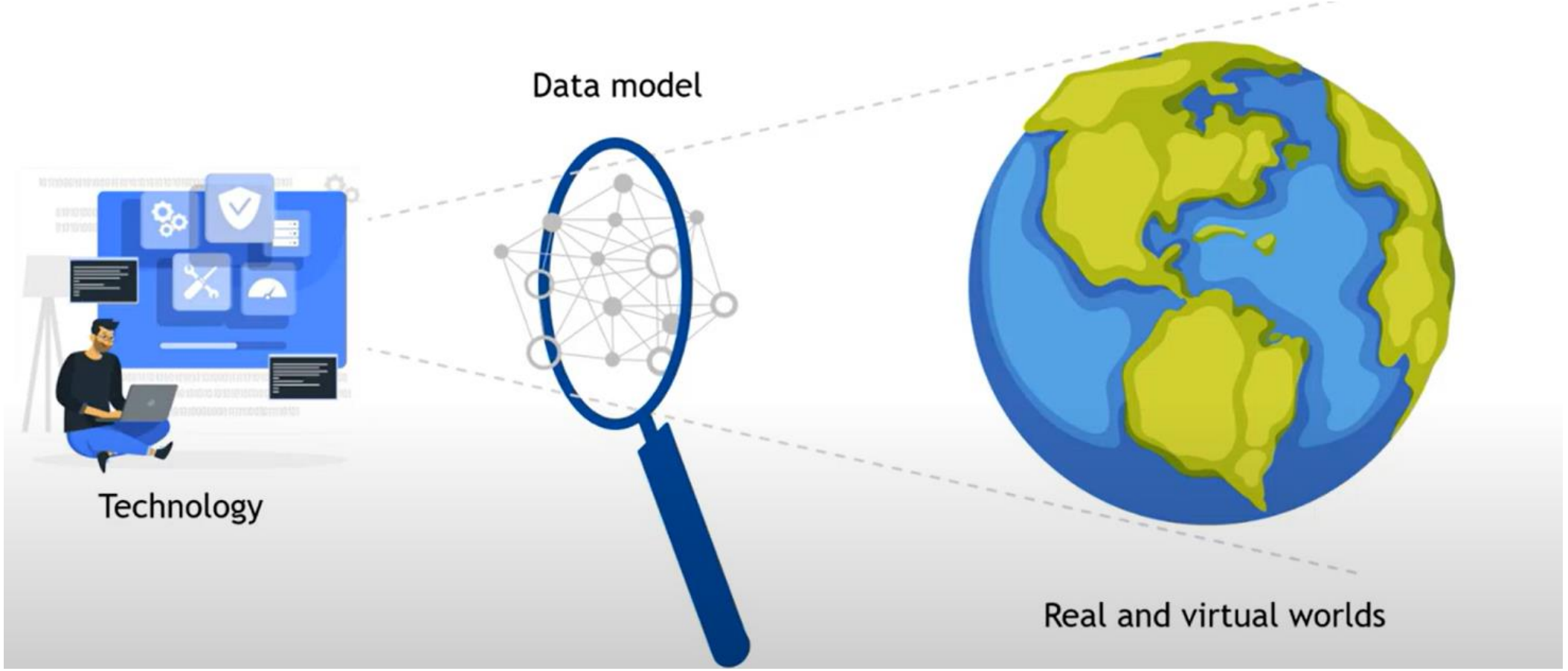
Software Architecture
Course's Code: CSE 483
Database Centric Architecture
(Chapter 5)

Chapter 5. Database Centric Architecture

Technology comes and goes, but data stays

Technology sees the world through data models



Data Aspect

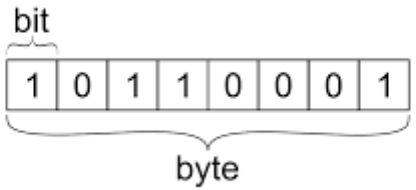
- Data can be **structured** or **non-structured**.
- **Structured** data follow a defined scheme like tables in a relational database or XML structures.

Customers

CustomerId	FirstName	LastName	DateCreated
1	Homer	Simpson	13/06/2014 3:33:37 PM
2	Peter	Griffin	13/06/2014 9:09:56 PM
3	Stewie	Griffin	13/06/2014 9:16:07 PM
4	Brian	Griffin	13/06/2014 9:16:36 PM
5	Cosmo	Kramer	13/06/2014 9:16:41 PM
6	Philip	Fry	13/06/2014 9:17:02 PM
7	Amy	Wong	13/06/2014 9:22:05 PM
8	Hubert J.	Farnsworth	13/06/2014 9:22:19 PM
9	Marge	Simpson	13/06/2014 9:22:37 PM
10	Bender	Rodriguez	13/06/2014 9:22:52 PM
11	Turanga	Leela	13/06/2014 9:23:37 PM
(New)			15/06/2014 9:00:01 PM

```
<studentsList>
    <student id="1">
        <firstName>Greg</firstName>
        <lastName>Dean</lastName>
        <certificate>True</certificate>
        <scores>
            <module1>70</module1>
            <module12>80</module12>
            <module3>90</module3>
        </scores>
    </student>
    <student ind="2">
        <firstName>Wirt</firstName>
        <lastName>Wood</lastName>
        <certificate>True</certificate>
        <scores>
            <module1>80</module1>
            <module12>80.2</module12>
            <module3>80</module3>
        </scores>
    </student>
</studentsList>
```

- **Non-structured** data are multimedia contents, e.g., images, audio, and video.

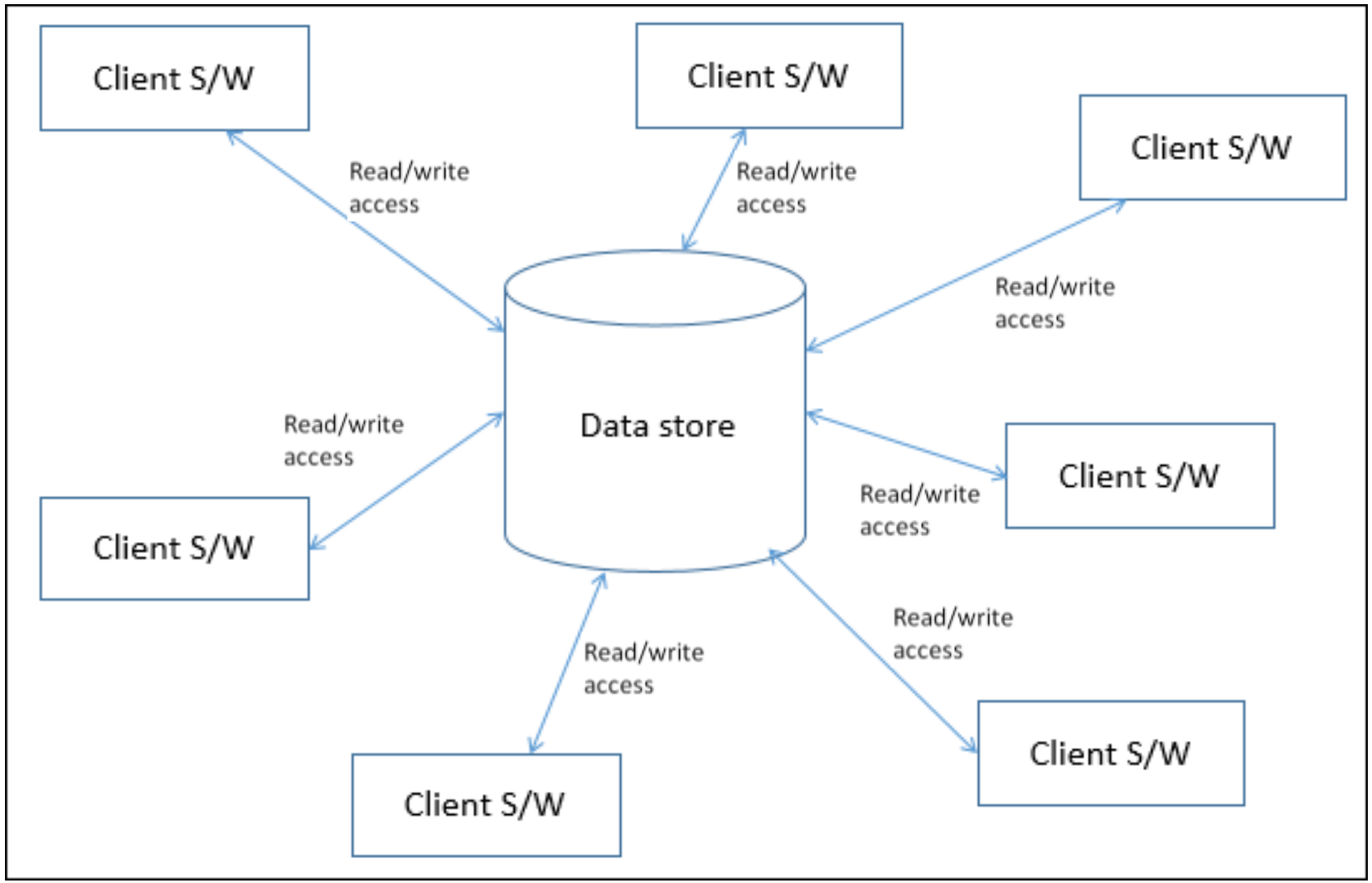


Data Grouping

Data can be grouped into 3 categories:

1. Structured data held in databases.
2. Document data in document management systems.
3. Multimedia data held in media servers.

Database Centric Architecture



Database Centric Architecture

A database centric architecture is based on the principle that processes /transactions communicate through a common/centralized repository which can be either passive or active.

Generally, it refers to architectures in which databases play a critical role

It may also mean a “relational database system” instead of an in-memory or file-based data structure.

the characterization of an architecture as "database-centric" may mean any combination of the following:

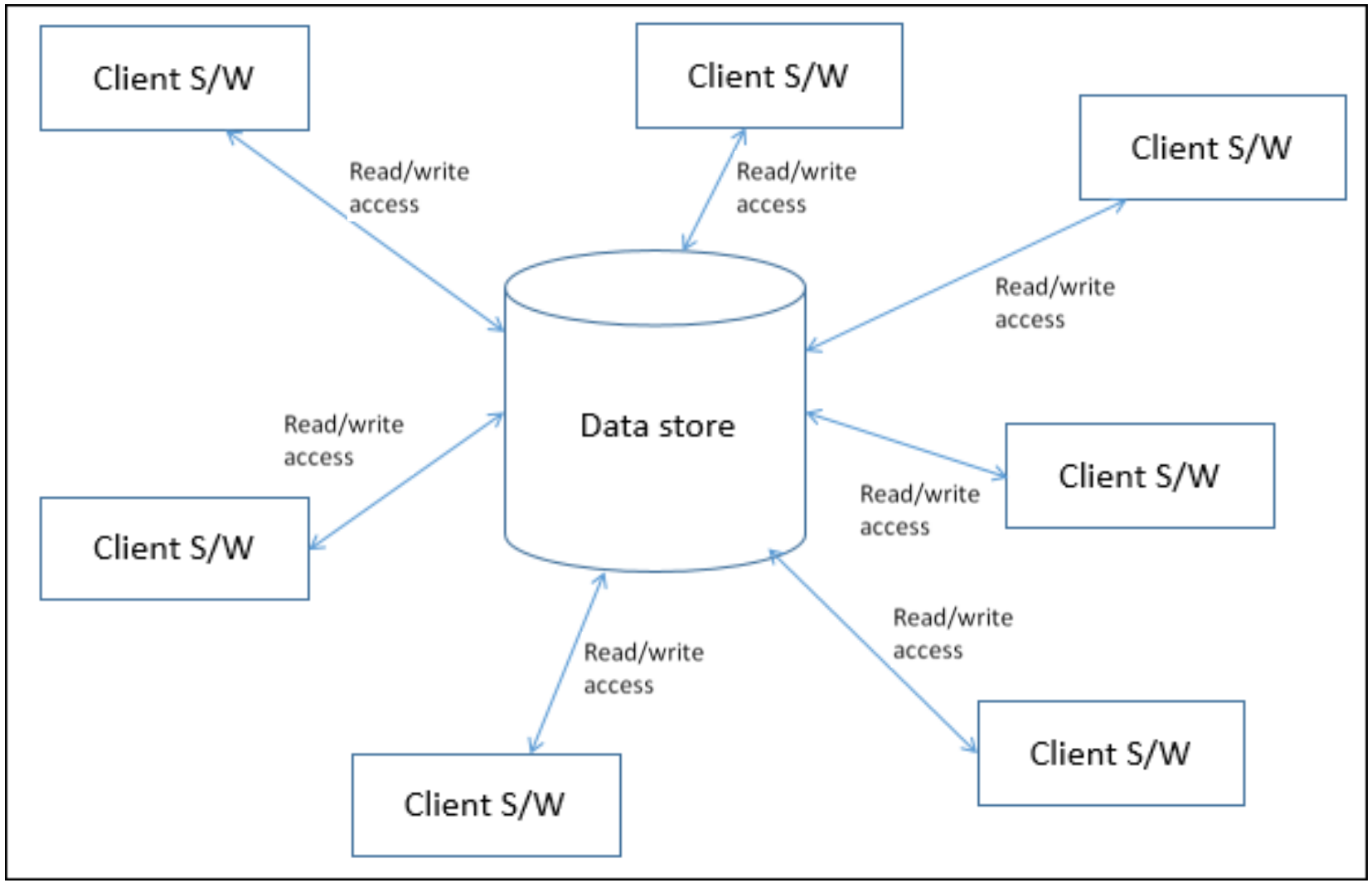
- Using a standard, general-purpose **relational database management system**, as opposed to customized in-memory or file-based data structures and access methods

- Using dynamic, **table-driven** logic - behavior that is heavily dictated by the contents of a database, allows programs to be simpler and more flexible.

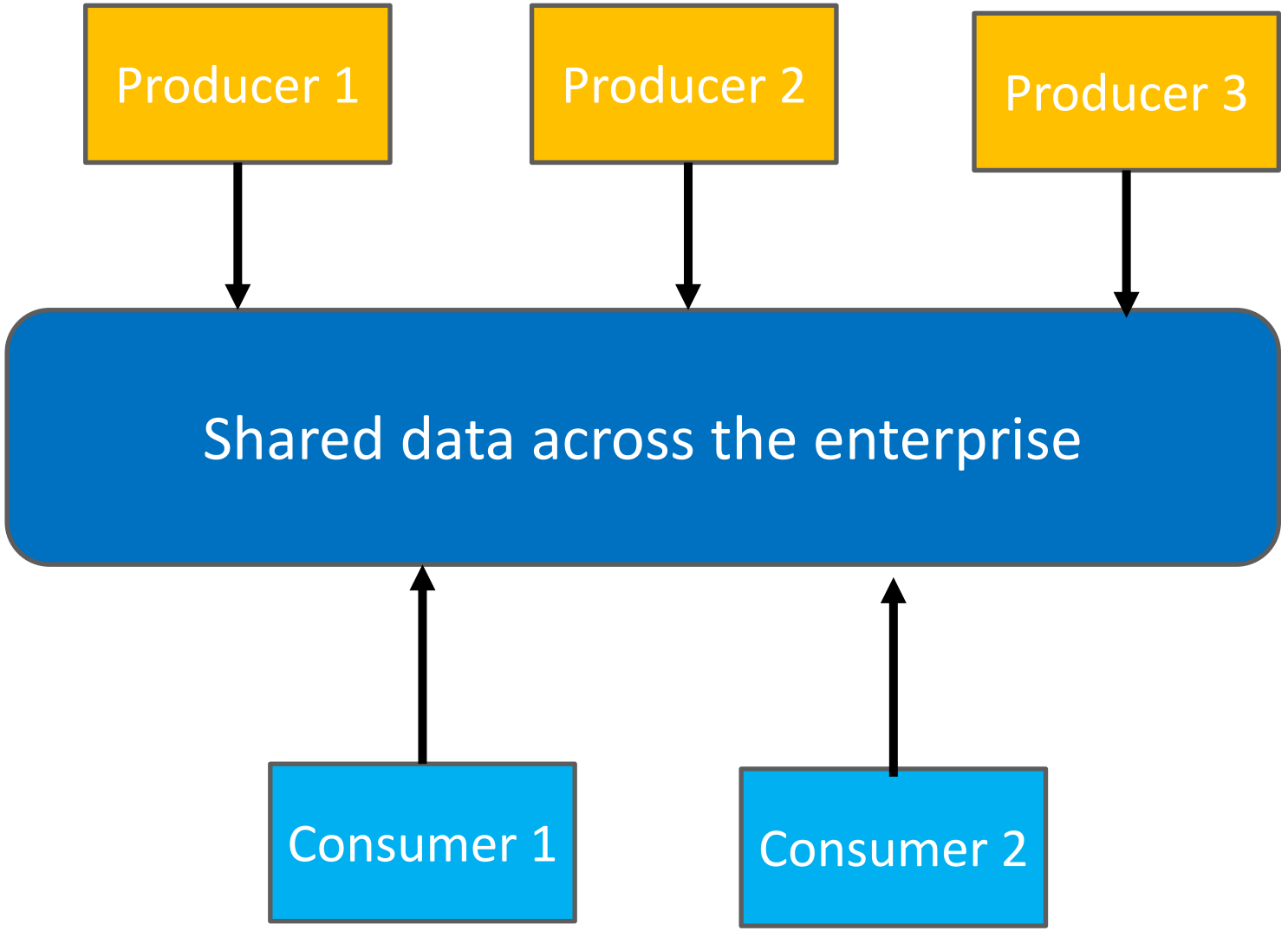
- Using **stored procedures** that run on database servers.

- Using a **shared database** as the basis for communicating between parallel processes in distributed computing applications.

Database Centric Architecture



The **producers of data** versus **consumers of data**



Informational requirements comes first

A good system architect is aware of the distinction between physical and logical concerns and will first want to ensure that the logical design of the system will fulfil the informational requirements for the enterprise before getting too concerned with implementation details:

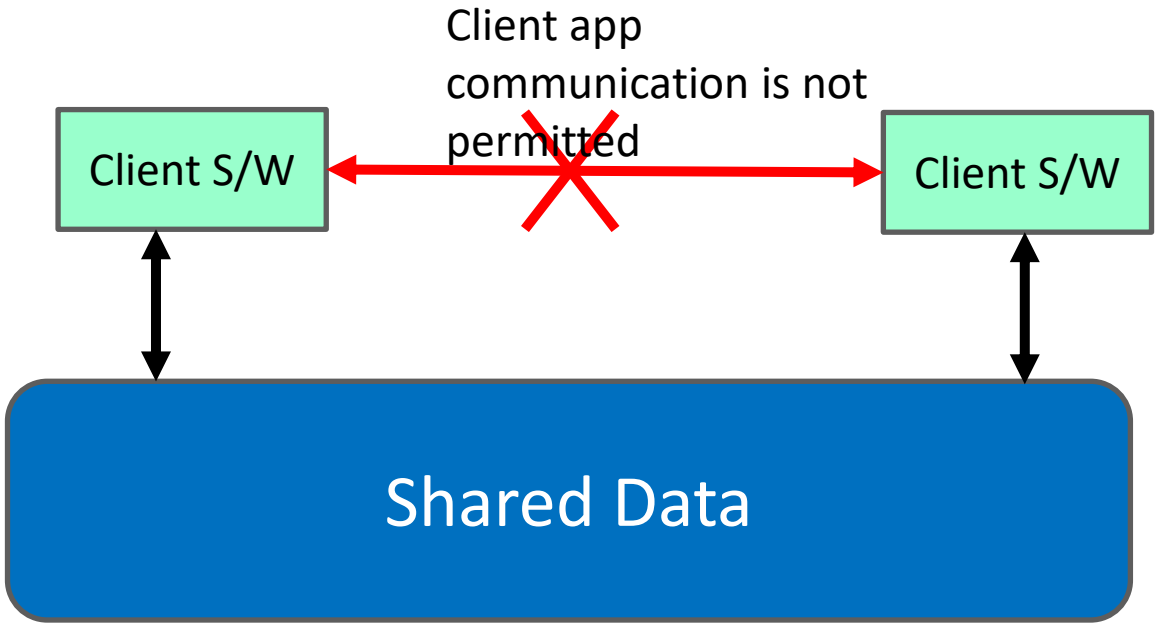
- Security
- Redundancy
- Backup
- Deployment
- Caching

Remember, logical requirements as a system comprised of shared data + applications

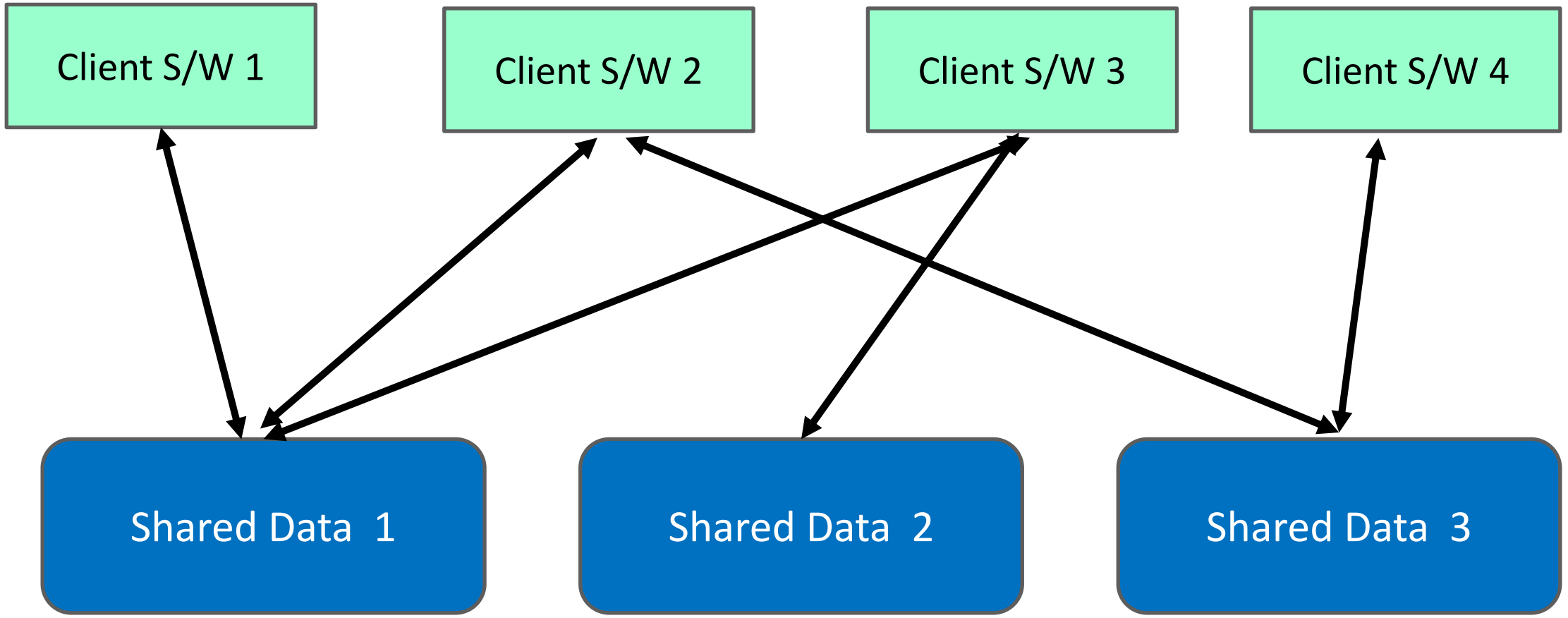
Applications

Applications can present the same data in many different ways.
New applications developed to browse, edit or present the existing data
Some applications can be disabled without affecting the database → decoupling of data and application at all scales.

There is no need for client applications to directly communicate with each other



Many relationship between data and applications



Characteristics of Data-Centric Architecture

Data Repositories: Data-centered architecture typically involves the establishment of centralized data repositories where all relevant data is stored. These repositories can be databases, data warehouses, data lakes, or other storage systems.

Data Modeling and Schema Design: The architecture focuses on designing robust data models and schemas that accurately represent the relationships and constraints within the data. This ensures data consistency and integrity.

Data Access and Manipulation: The architecture emphasizes the design of mechanisms for accessing, querying, and modifying data stored in the repositories. This may involve the use of standardized query languages, APIs, or other interfaces.

Data Security and Privacy: Ensuring the security and privacy of sensitive data is a crucial aspect of data-centered architecture. Access controls, encryption, and authentication mechanisms are often integrated to protect the data.

Characteristics of Data-Centric Architecture

Data Sharing and Integration: Data-centered architectures facilitate the sharing and integration of data across different components and services within an application or across different applications. This is particularly important in systems with diverse functionalities.

Data Consistency and Coherency: Maintaining data consistency and coherency across different parts of the system is a priority in data-centered architecture. Techniques like transactions and synchronization mechanisms may be employed.

Scalability and Performance: The architecture needs to address scalability concerns, especially when dealing with large volumes of data. Techniques like sharding, replication, and caching may be utilized to optimize performance.

Analytics and Reporting: Data-centered architectures often include provisions for data analysis, reporting, and generating insights from the stored data.

Benefits of Data-Centric Architecture

Data Integrity: By making data a central focus, the architecture ensures that data remains consistent, accurate, and reliable throughout the system.

Modularity: Different components can access and manipulate data without tightly coupling with each other, promoting modularity and maintainability.

Reusability: Data-centric services and data models can be reused across different parts of the application or in different applications.

Scalability: Data stores and data access mechanisms can be scaled independently to handle increased data volume and user demands.

Benefits of Data-Centric Architecture

Consistency: All components and services work with the same set of data, reducing the chances of discrepancies or conflicting information.

Ease of Integration: Data-centered architectures simplify the integration of different modules or services that need to share and exchange data.

Centralized Management: Centralized data repositories make it easier to manage and control data-related aspects of the system.

Use Cases of Data-Centric Architecture

Healthcare Systems: Electronic health record systems and patient data management benefit from the organization and security provided by data-centered architecture.

Content Management Systems: Systems managing and delivering content across websites rely heavily on well-organized data structures.

Chapter 5. Database Centric Architecture

5.1 What is database-centric architecture?

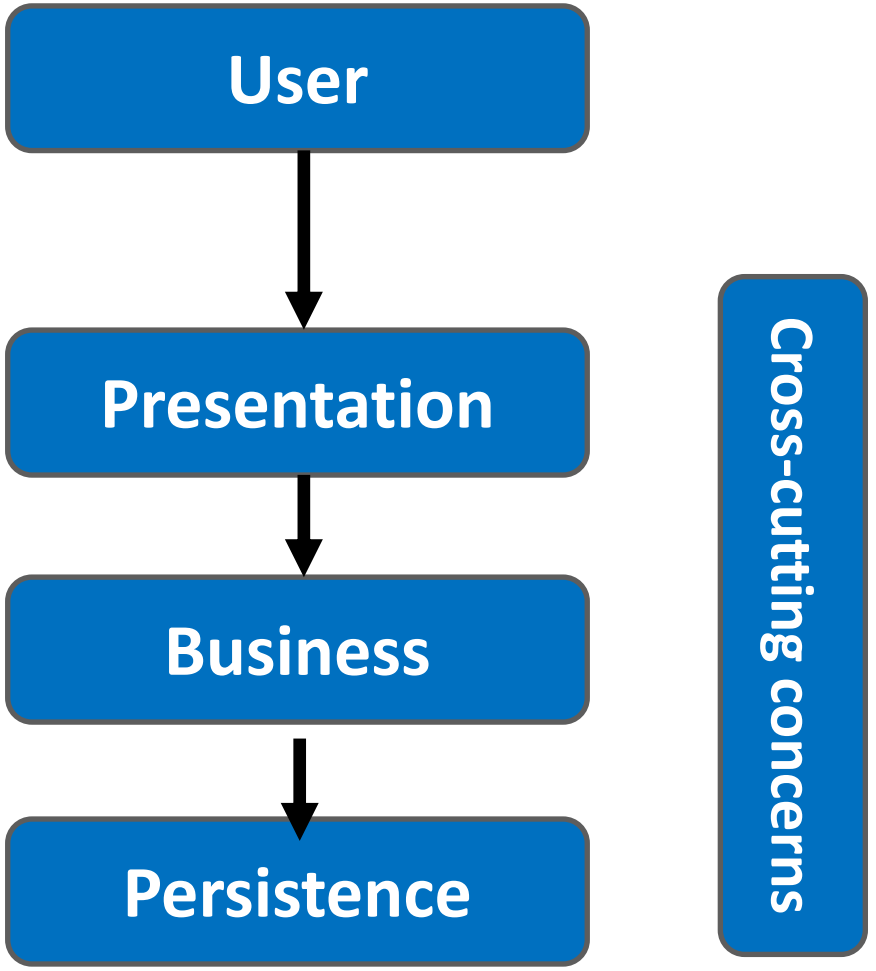
5.2 Classic three-tier architecture?

5.3 Advantages and disadvantages

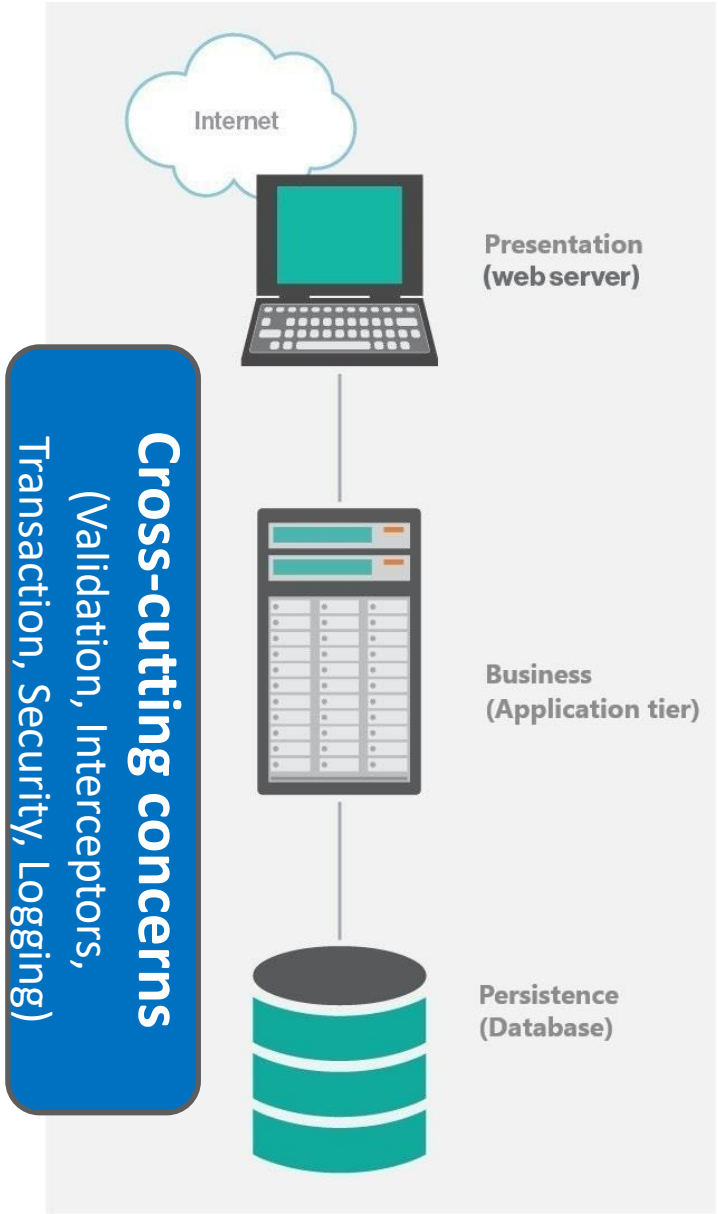
Classic Three-Tier Architecture

Three-Tier Architecture is a well-established software application design pattern which will organizes the application in the **three logical and physical computing tiers** as following:

- **Presentation Tier:** user interface
- **Business (Application Tier):** handles business logic
- **Persistence (Data Tier):** stores information



Classic Three-Tier Architecture



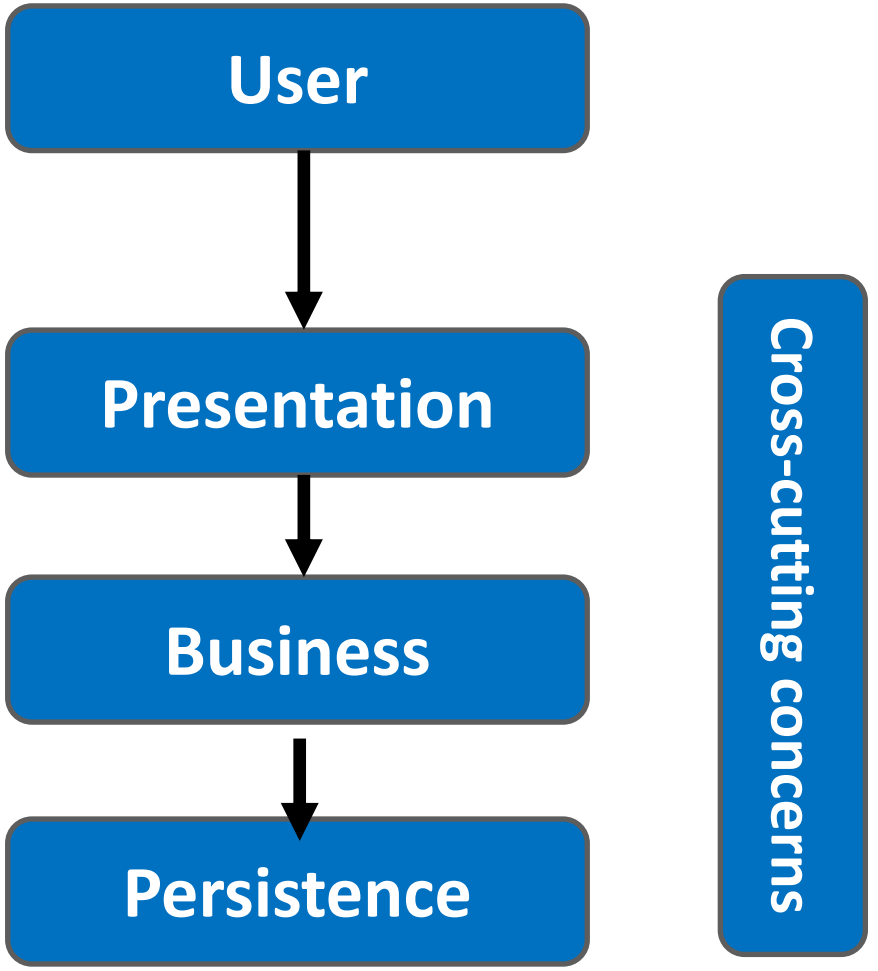
Presentation tier: This tier, which is built with HTML5, cascading style sheets (CSS) and JavaScript, is deployed to a computing device through a web browser or a web-based application. The presentation tier communicates with the other tiers through application program interface (API) calls.

Application tier (Business): The application tier, which may also be referred to as the logic tier, is written in a programming language such as Java and contains the business logic that supports the application's core functions. The underlying application tier can either be hosted on distributed servers in the cloud or on a dedicated in-house server, depending on how much processing power the application requires.

Data tier (Persistence): The data tier consists of a database and a program for managing read and write access to a database. This tier may also be referred to as the storage tier and can be hosted on-premises or in the cloud. Popular database systems for managing read/write access include MySQL, PostgreSQL, Microsoft SQL Server and MongoDB.

Classic Three-Tier Architecture

- **Very widely used**
- **Build monolithic application**



Chapter 5. Database Centric Architecture

5.1 What is database-centric architecture?

5.2 Classic three-tier architecture?

5.3 Advantages and disadvantages

Advantages and Disadvantages

Advantages

Well-known and familiar

Simple to reason about

Tooling design for three-tier

Created for CRUD (Create, Read, Update, Delete)

Advantages and Disadvantages

Disadvantages

- Difficulties with complex domains
- Unclear abstraction
- Layer coupling