Lecture05: Distributed System Architecture

EGCI341

Outline

- Distributed system
- Layered Application architecture
  - Thin and Fat Clients
  - Three-Tier architecture
  - Distributed-object architecture
- Application Architectures

Distributed Systems

- Virtually all large computer-based systems are now ***distributed systems***
- Information processing is distributed over several computers rather than confined to a single machine
- Distributed software engineering is very important for enterprise computing systems

"Can you give me an example of distributed system"?"

Distributed System Characteristics

- Resource sharing
  - Sharing of hardware and software resources
- Openness
  - Use of equipment and software from different vendors
- Concurrency
  - Concurrent processing to enhance performance
- Scalability
  - Increased throughput by adding new resources
- Fault tolerance
  - The ability to continue in operation after a fault has occurred

Distributed System Disadvantages

- Complexity
  - Distributed systems are more complex than centralized systems
- Security
  - More susceptible to external attack
- Manageability
  - More effort required for system management
- Unpredictability
  - Unpredictable responses depending on the system organization and network load

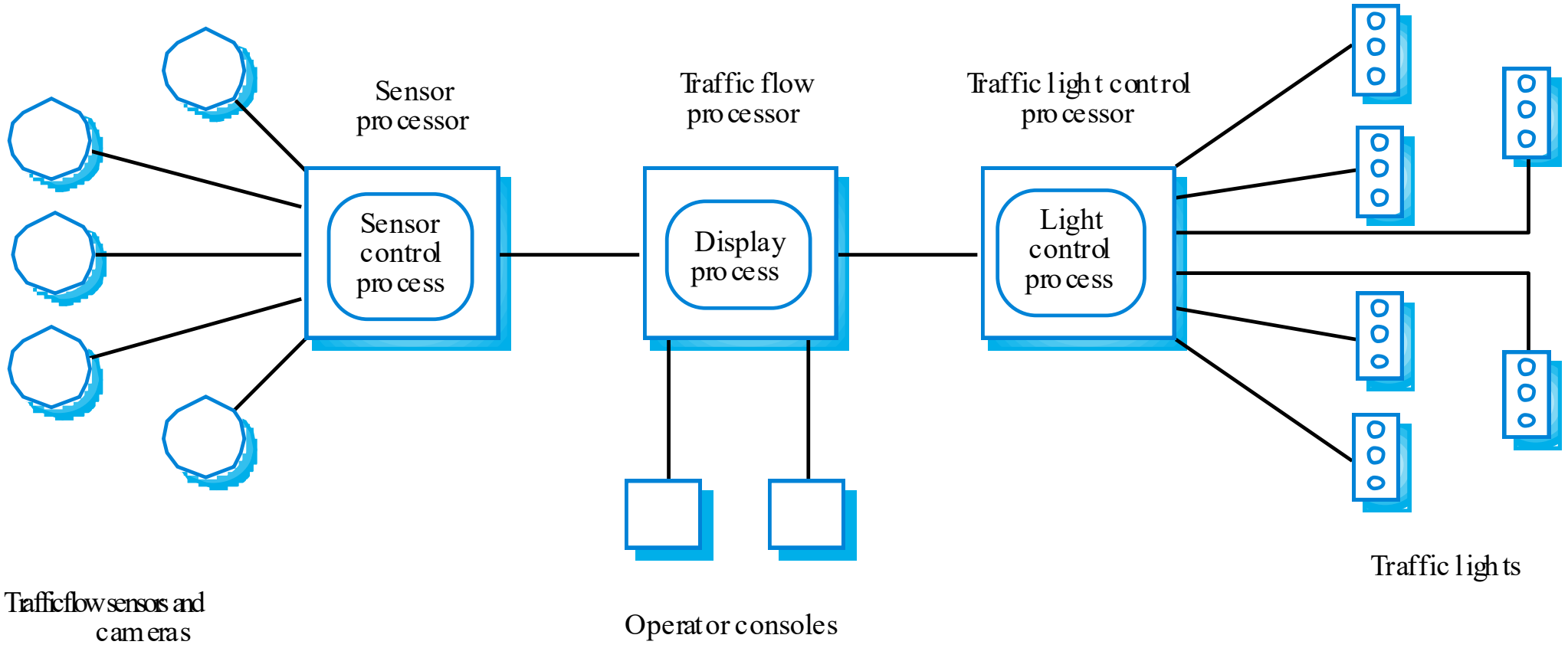
Distributed Systems Architectures

- Client-server architectures
  - Distributed services which are called on by clients
  - Servers that provide services are treated differently from clients that use services
- Distributed object architectures
  - No distinction between clients and servers
  - Any objects on the system may provide and use services from other objects

Multiprocessor Architectures

- Simplest distributed system model
- System composed of multiple processes which can execute on different processors
- Architectural model of many large real-time systems
- Distribution of processor to processor may be pre-ordered or may be under the control of a dispatcher

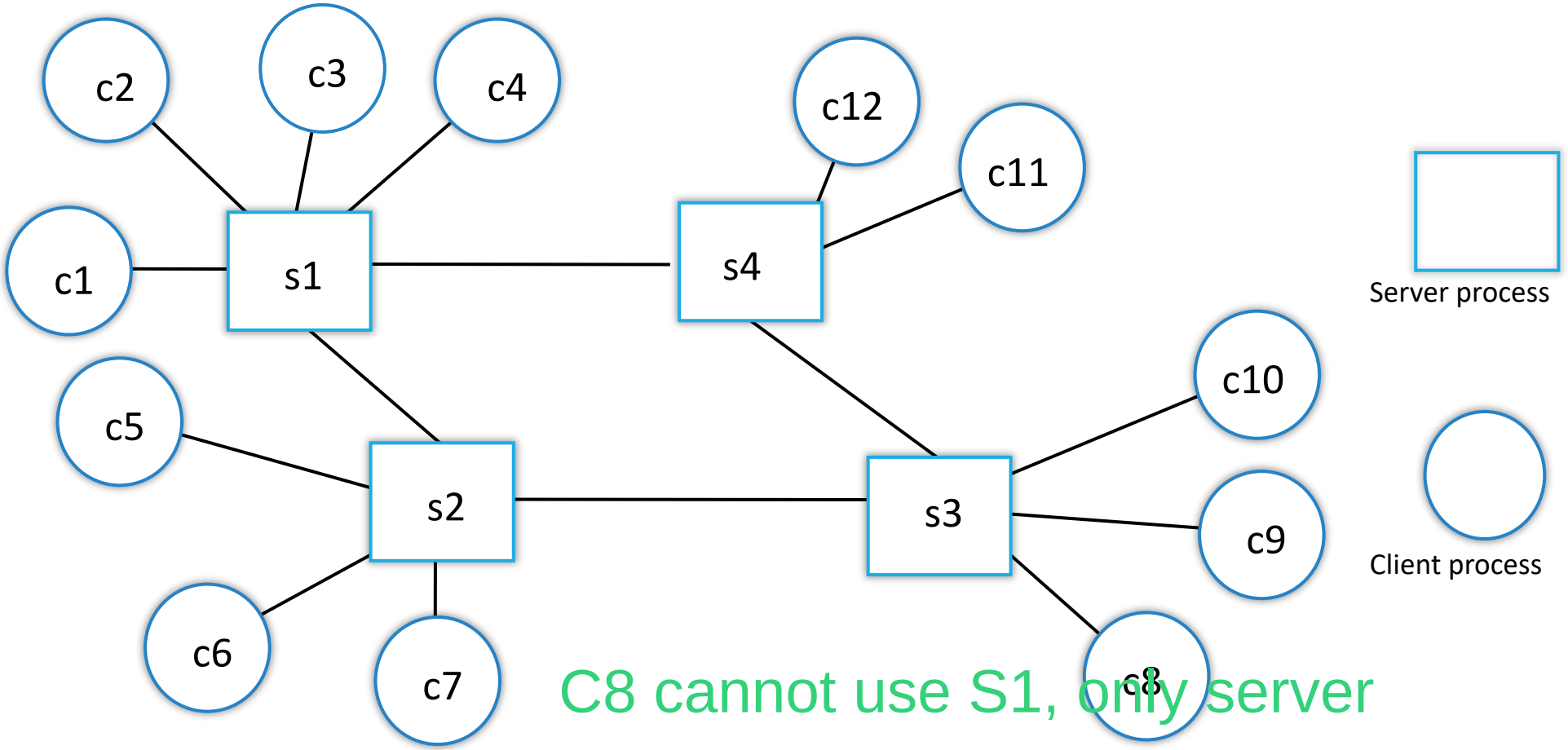
A Multiprocessor Traffic Control System [1]



Client-Server Architectures

- Application is modeled as a set of services and provided by ***servers and a set of clients*** that use these services
- Clients know of servers but servers do not need to know of clients
- Mapping of processors to processes is not necessarily 1 : 1

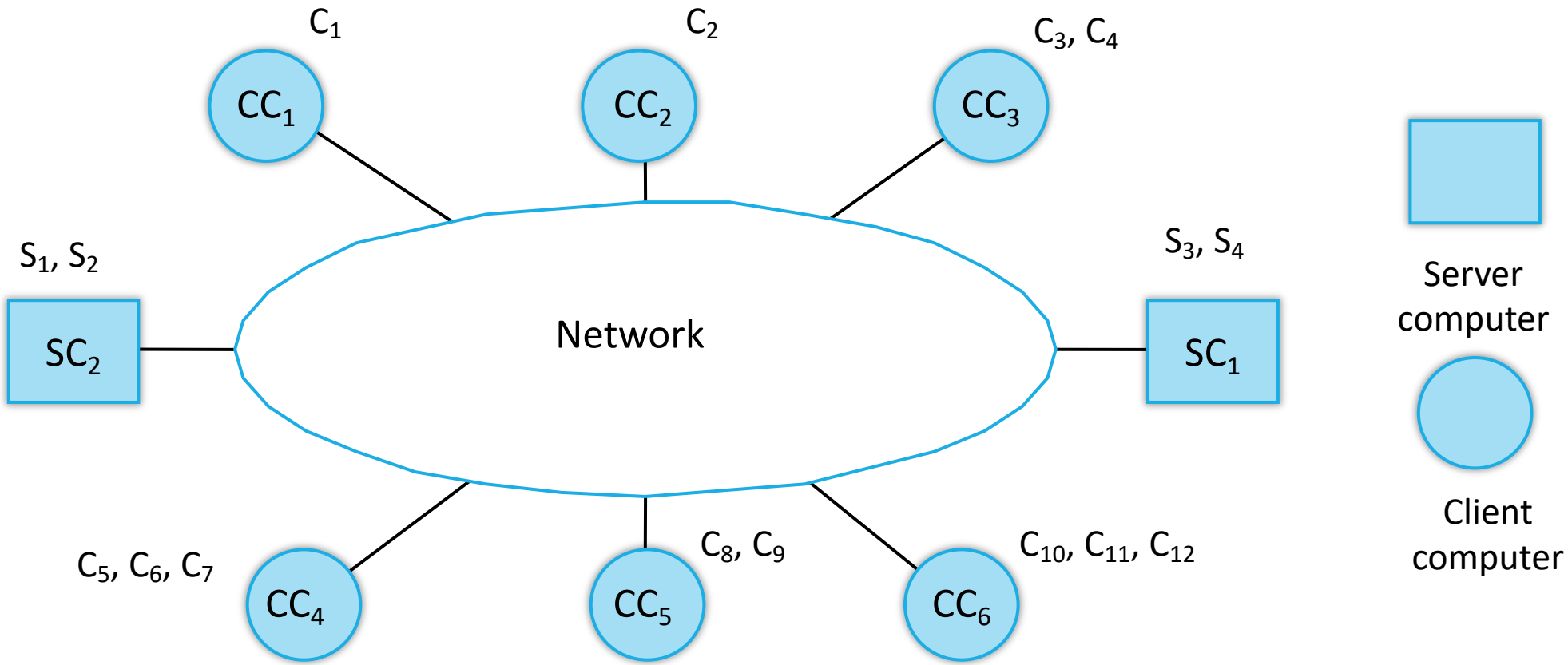
Client-Server System



C8 cannot use S1, only server can communicate with each other

Computers in a C/S network

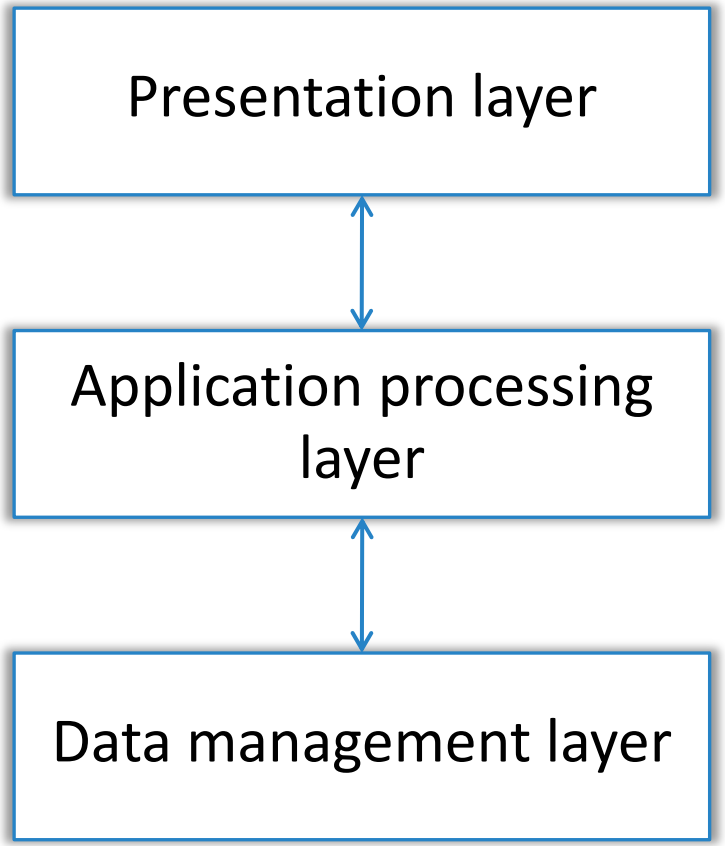
This one you, every client can use every server



Layered Application Architecture

1. **Presentation layer** Front-end : html, css
   - Concerned with presenting the results of a computation to system users and with collecting user inputs
2. **Application processing layer** eg. PHP code for functionallity
   - Concerned with providing application specific functionality
   - Example: In a banking system, banking functions such as open account, close account, etc.
3. **Data management layer** Database server
   - Concerned with managing the system databases

Application Layers



Thin and Fat Client Model

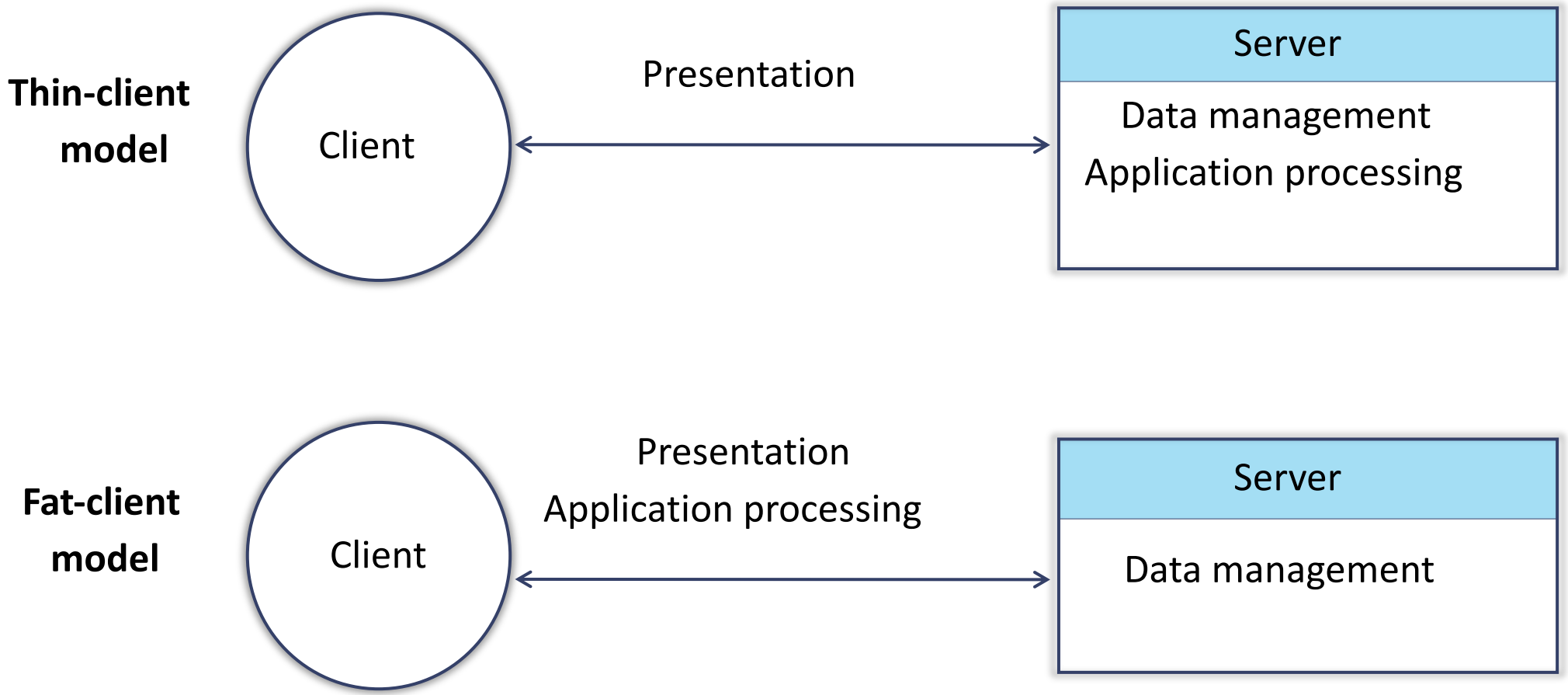
Thin-client model

- All of the application processing and data management is carried out on the server
- Client is simply responsible for running the presentation software

Fat-client model

- Server is only responsible for data management
- Software on the client implements the application logic and the interactions with the system user

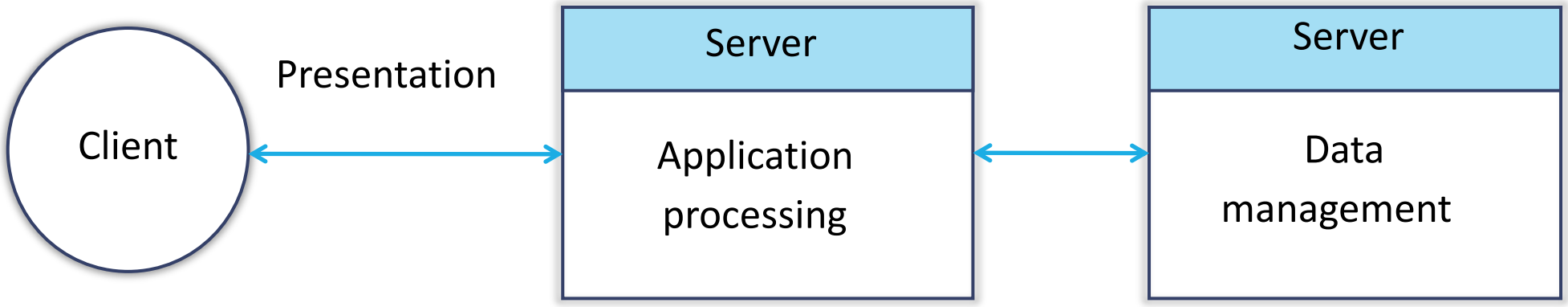
Thin and Fat Client Model (Cont.)



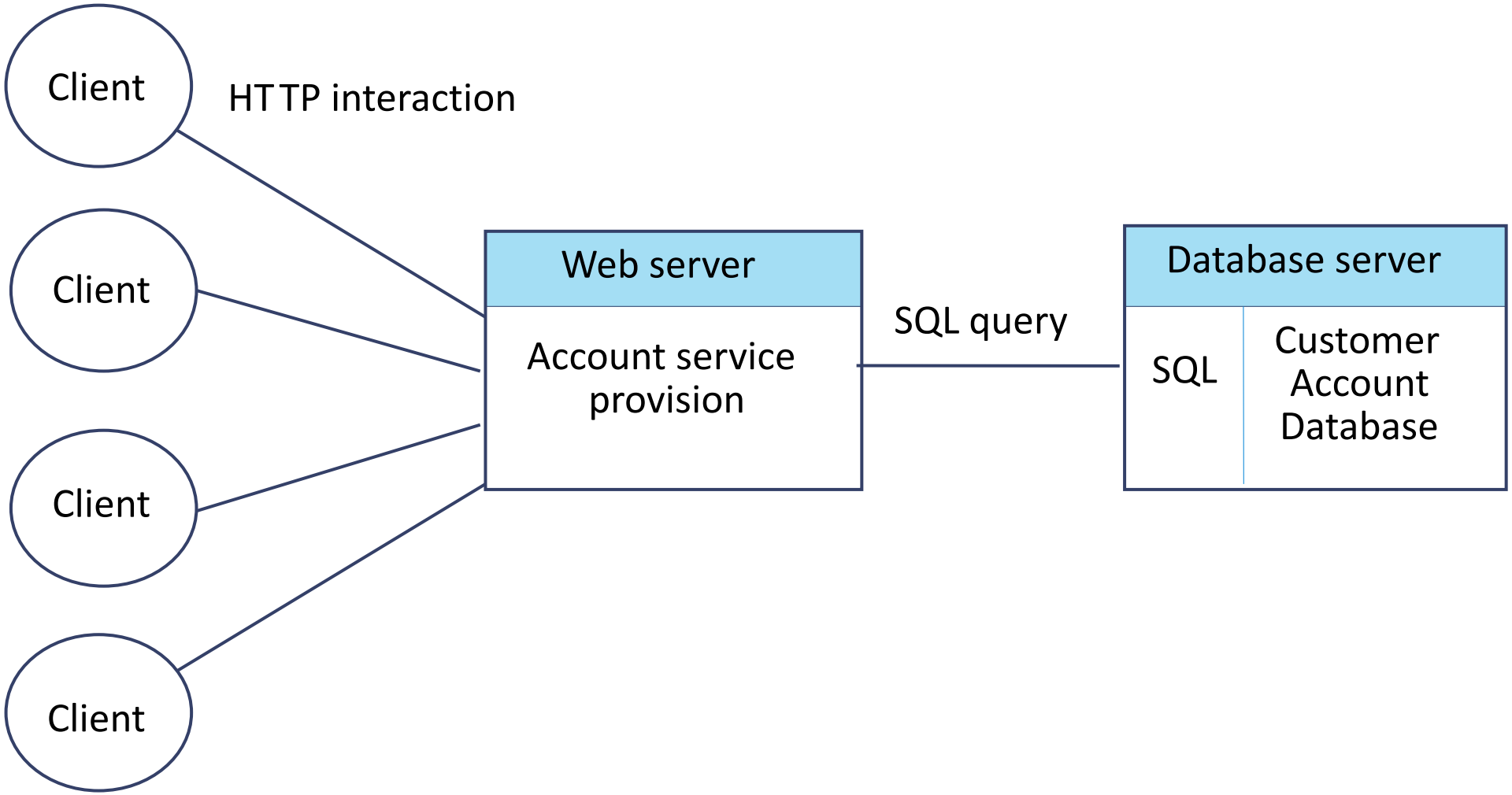
Three-Tier Architecture

- Each of the application architecture layers may execute on a separate processor
- Allow for better performance than a thin-client approach and it is simpler to manage than a fat-client approach
- A more scalable architecture - as demands increase, extra servers can be added -

A Three-tier Client-Server Architecture



Internet Banking System



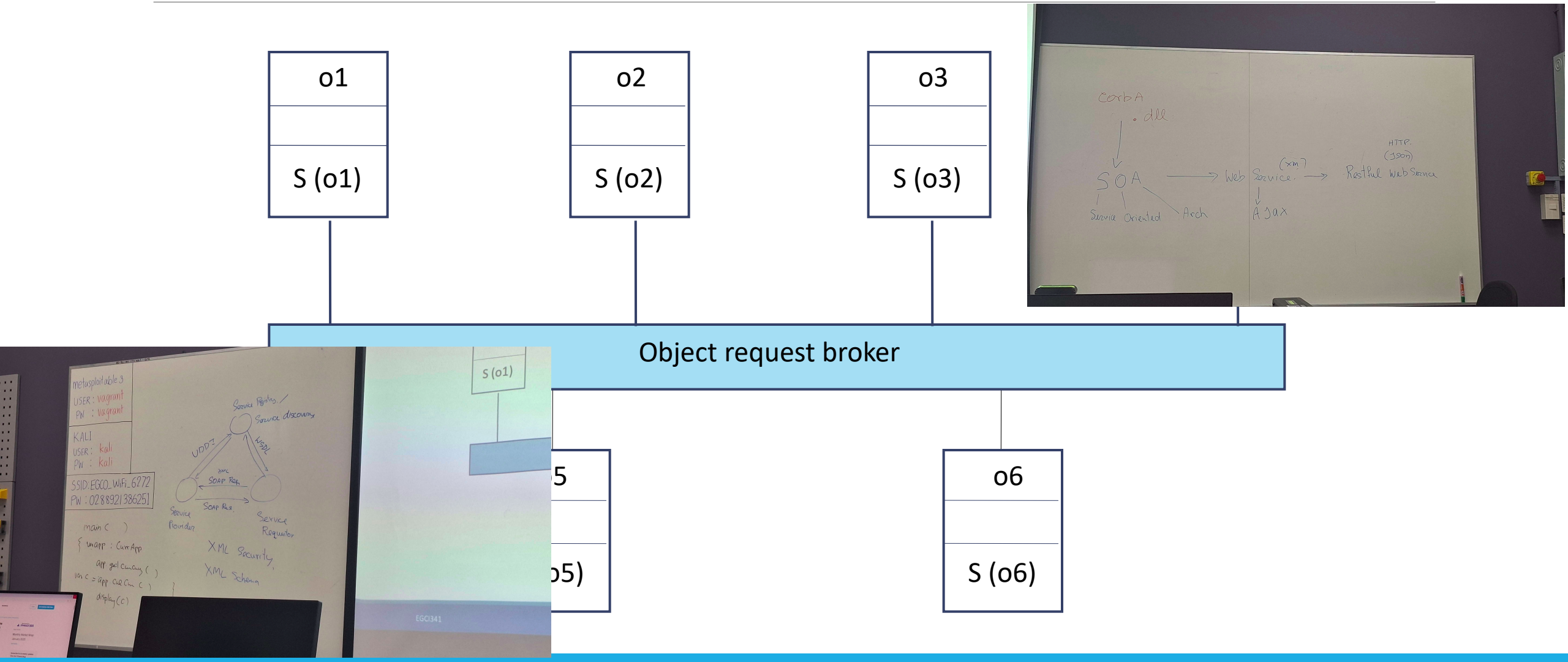
Use of Client-Server Architecture

Architecture	Applications
Two-tier C/S architecture with thin clients	Legacy system applications where separating application processing and data management is impractical. Computationally-intensive applications such as compilers with little or no data management. Data-intensive applications (browsing and querying) with little or no application processing.
Two-tier C/S architecture with fat clients	Applications where application processing is provided by off-the-shelf software (e.g. Microsoft Excel) on the client. Applications where computationally-intensive processing of data (e.g. data visualisation) is required. Applications with relatively stable end-user functionality used in an environment with well-established system management.
Three-tier or multi-tier C/S architecture	Large scale applications with hundreds or thousands of clients Applications where both the data and the application are volatile. Applications where data from multiple sources are integrated.

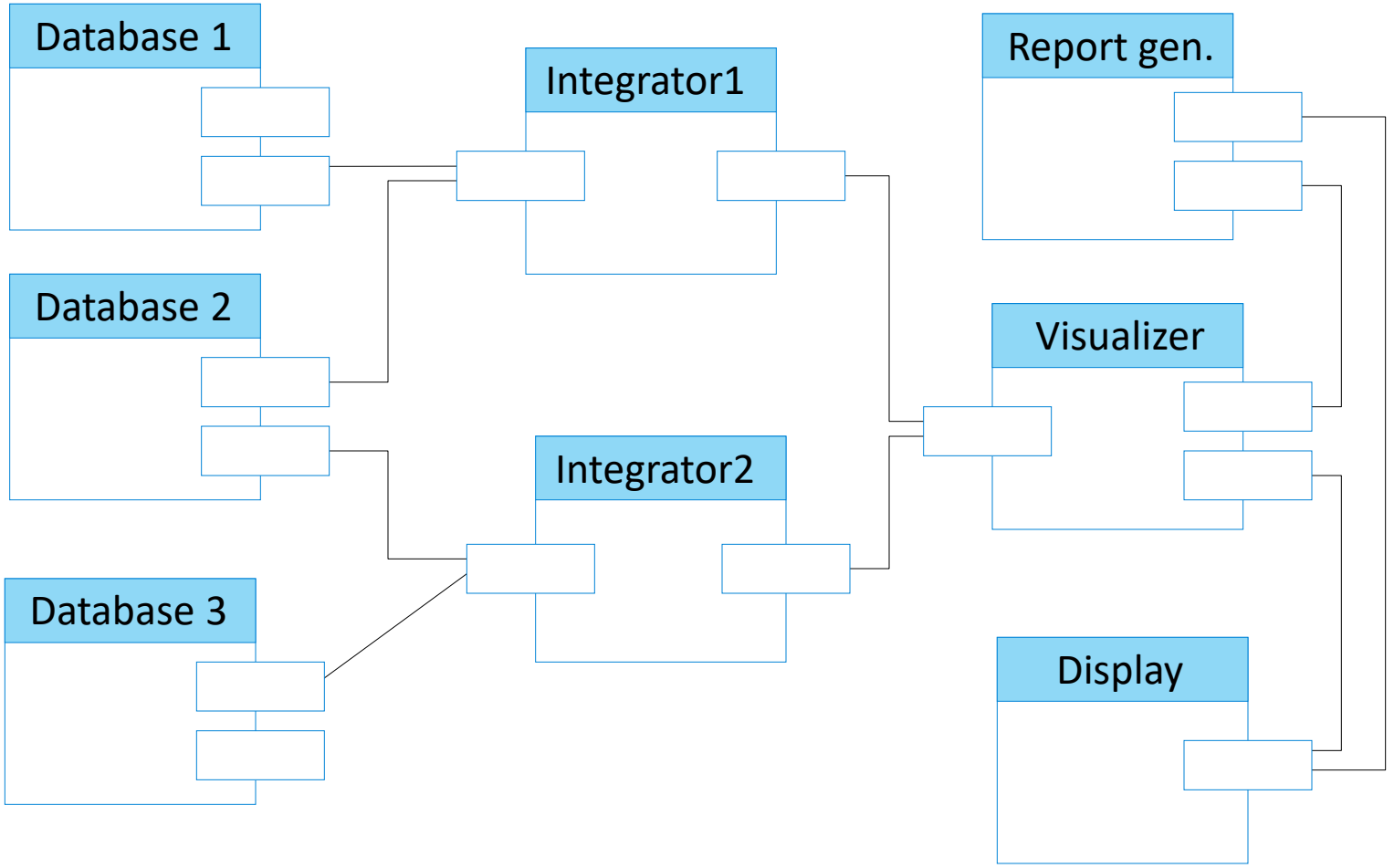
Distributed Object Architecture

- There is no distinction in a distributed object architectures between clients and servers
- Each distributable entity is an object that provides services to other objects and receives services from other objects
- Object communication is through a middleware system called an object request broker
- However, distributed object architectures are more complex to design than C/S systems

Distributed Object Architecture (Cont.)



Data Mining System



Application Architectures

Application Types

1. Data processing applications
   - Applications process data in batches without explicit user intervention during the processing
2. Transaction processing applications Booking system, SKY+
   - Applications process user requests and update information in a system database
3. Event processing systems React to event : Games
   - System actions depend on interpreting events from the system's environment
4. Language processing systems
   - Applications where the users' intentions are specified in a formal language that is processed and interpreted by the system

Application Type Examples

Data processing systems

- Billing systems
- Payroll systems

Transaction processing systems

- E-commerce systems
- Reservation systems

Event processing systems

- Word processors
- Real-time systems

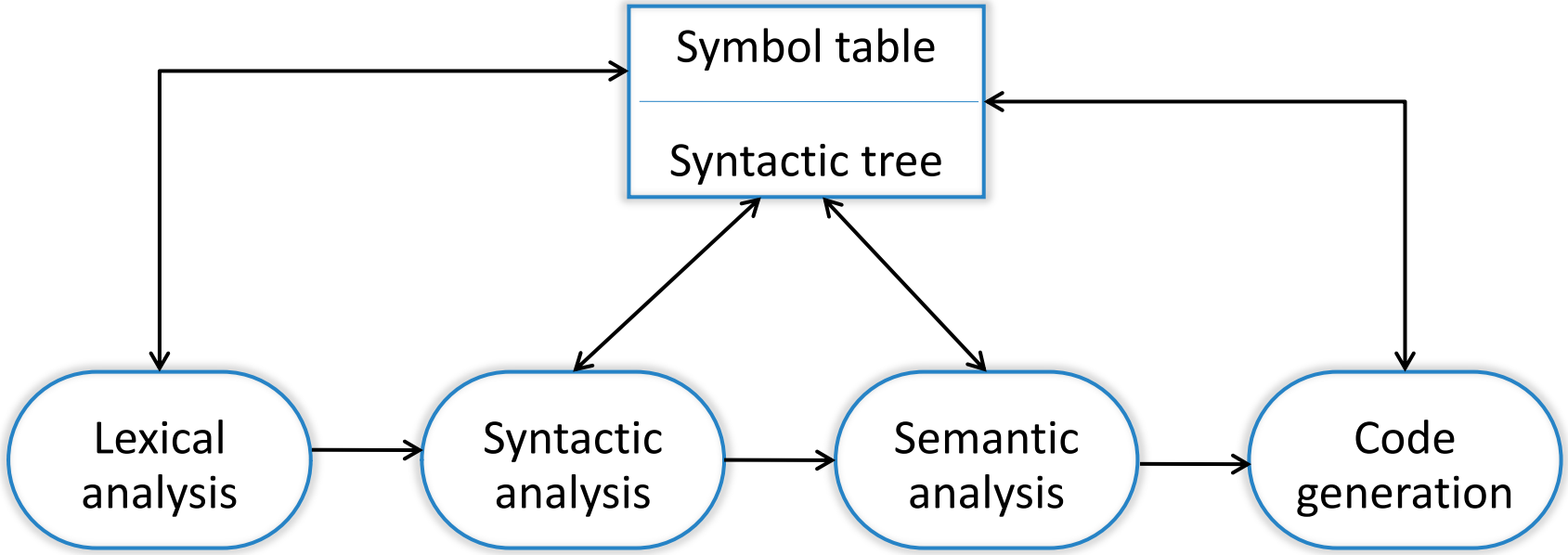
Language processing systems

- Compilers
- Command interpreters

1. Data Processing Systems

- Data is input and output in batches
  - **Input:** A set of customer numbers and associated readings of an electricity meter
  - **Output:** A corresponding set of bills, one for each customer number

Data-Flow Model of a Compiler

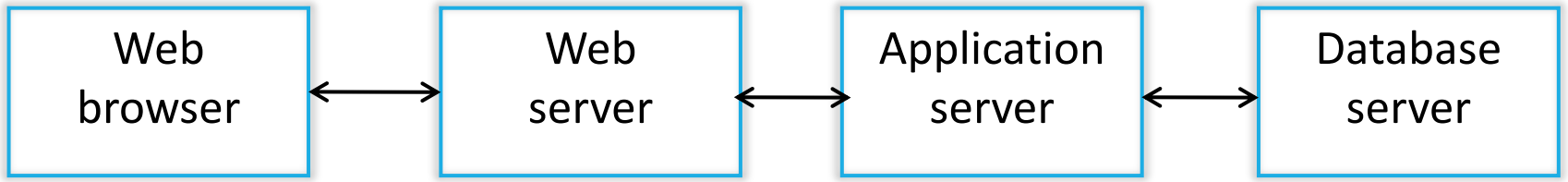


2. Transaction Processing Systems

- System processes user requests for information from a database or requests to update the database
- From a user perspective, a **transaction** is:
  - Any coherent sequence of operations that satisfies a goal
  - For example - find the times of flights from London to Paris
- Users make asynchronous requests for service which are then processed by a transaction manager

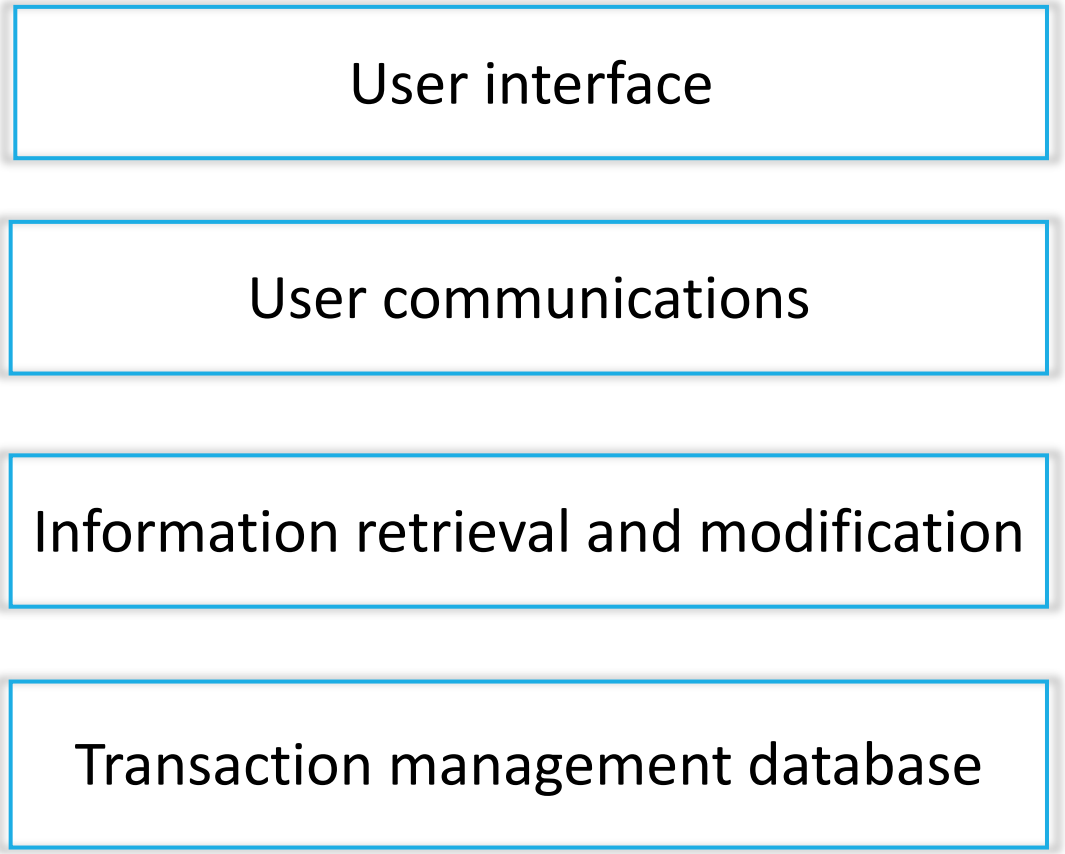
E-Commerce System Architecture

- E-commerce systems are Internet-based resource management systems that accept electronic orders for goods or services.
- They are usually organized using a multi-tier architecture with application layers associated with each tier



Information System Architecture

- Information systems have a generic architecture that can be organized as a layered architecture
- Layers include:
  - User interface
  - User communications
  - Information retrieval
  - System database



3. Event Processing Systems

- Systems respond to events in the system's environment
- Their key characteristic is that event timing is unpredictable so the architecture has to be organized to handle this
- Many common systems are event processing systems, such as word processors, games, etc.

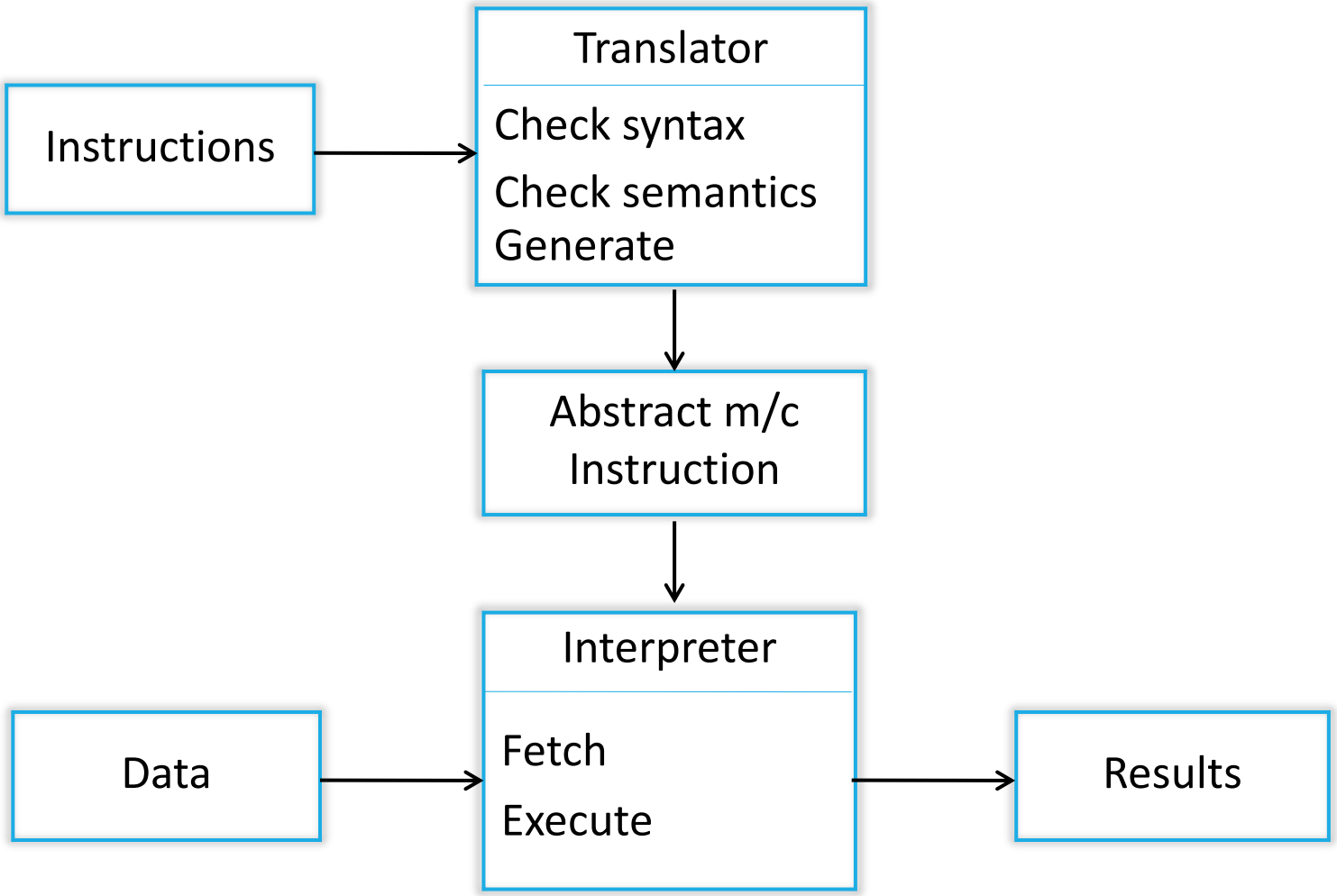
Editing Systems

- Real-time system and editing system are the most common types of event processing system
- Editing system characteristics:
  - Single user systems
  - Must provide rapid feedback to user actions
  - Organize long transactions so may include recovery facilities

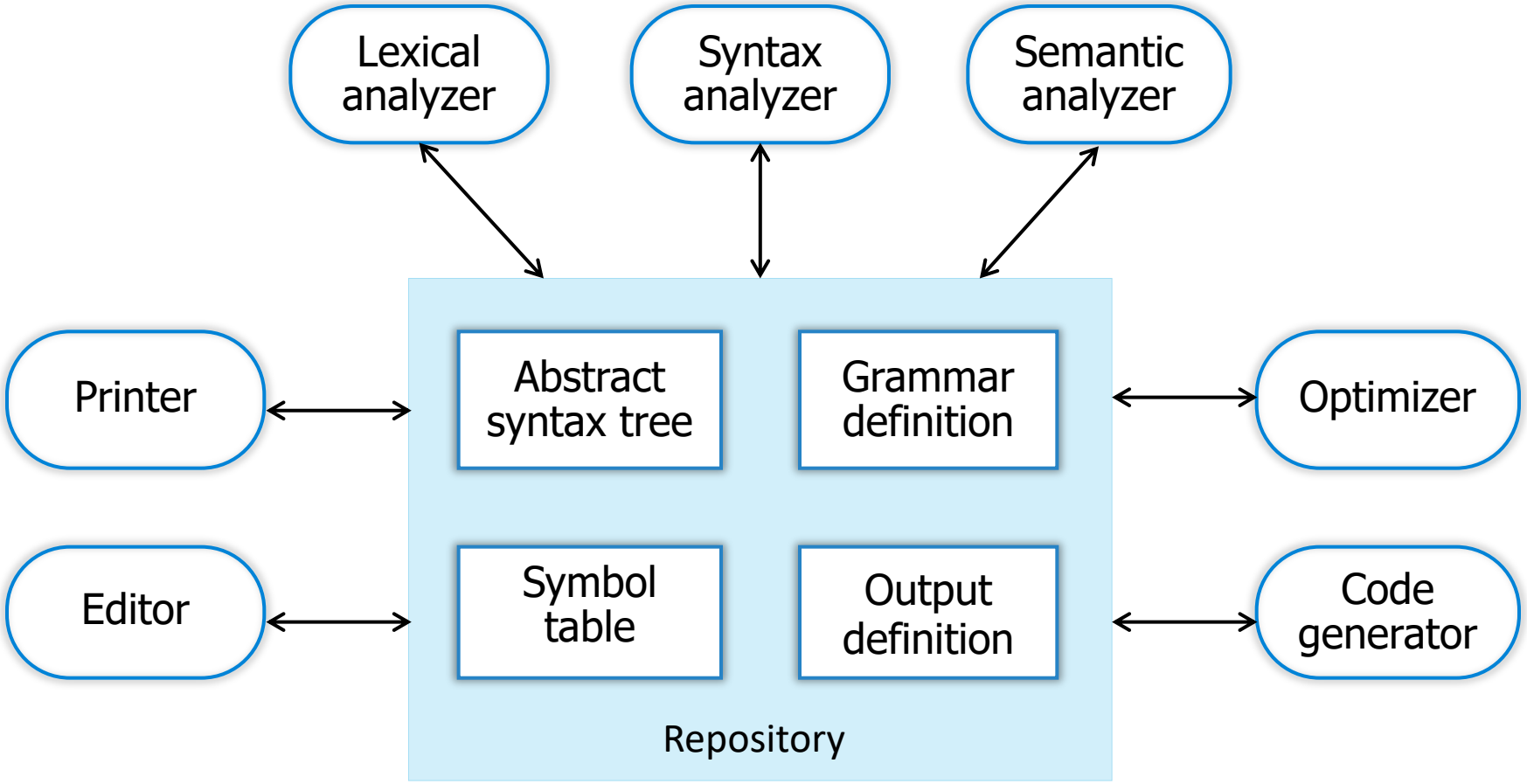
4. Language Processing Systems

- Accept a natural or artificial language as input and generate some other representation of that language
- May include an interpreter to act on the instructions in the language that is being processed
- Used in situations where the easiest way to solve a problem is to describe an algorithm or describe the system data
  - Meta-case tools process tool descriptions, method rules and generate tools

Language Processing System



Repository Model of Compiler



Reference

- Ian Sommerville, Software Engineering 10th Edition, Addison-Wesley, April 2015

Any Questions?

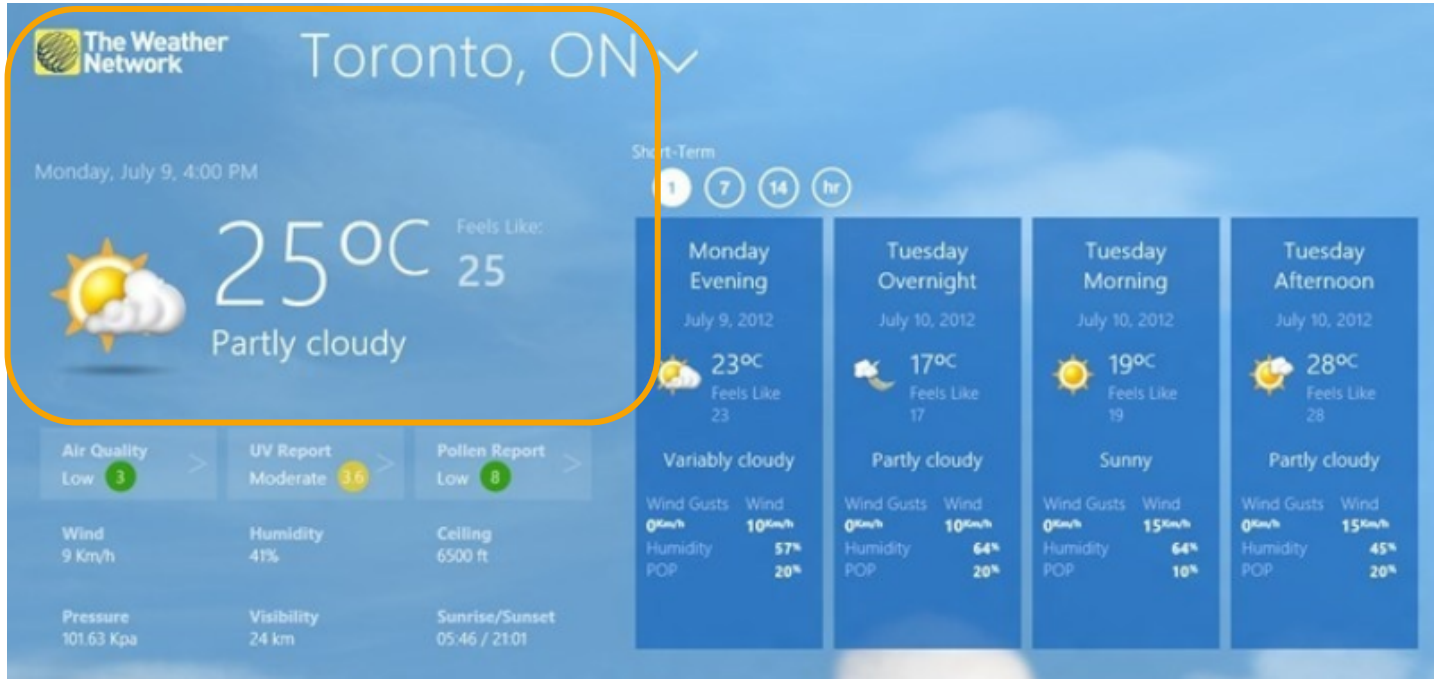
:O)

Thank you

Practice: Software Architecture Design (1)

Design the architecture for the application below

- The weather report website
  - Assume that the weather data are ready in the database, the application can retrieve the weather information according to the city name and date. These information can be reported on the website (online).



Ref. https://www.theweathernetwork.com/

Practice: Software Architecture Design (2)

Design the architecture for the application below

- Currency Exchange Mobile Application
  - Assume that the exchange rate can be retrieved from currency web service (online). The application converts the money by submitting the amount of money and currency from both sides