COMPSCI 320 Introduction to Software Engineering:

Architecture/High Level Design

Slides adapted from Gordon Anderson

One-Slide Summary

- **Architecture** helps us plan how **components** of a system are **connected**

- Architectural **views** let us describe different the system at different levels of granularity

- Good software architecture considers:
  - Separation of **Interface** from **Implementation**
  - **Coupling** (amount of dependence) between components
  - Division of **responsibilities** across components

Common Architectural Views

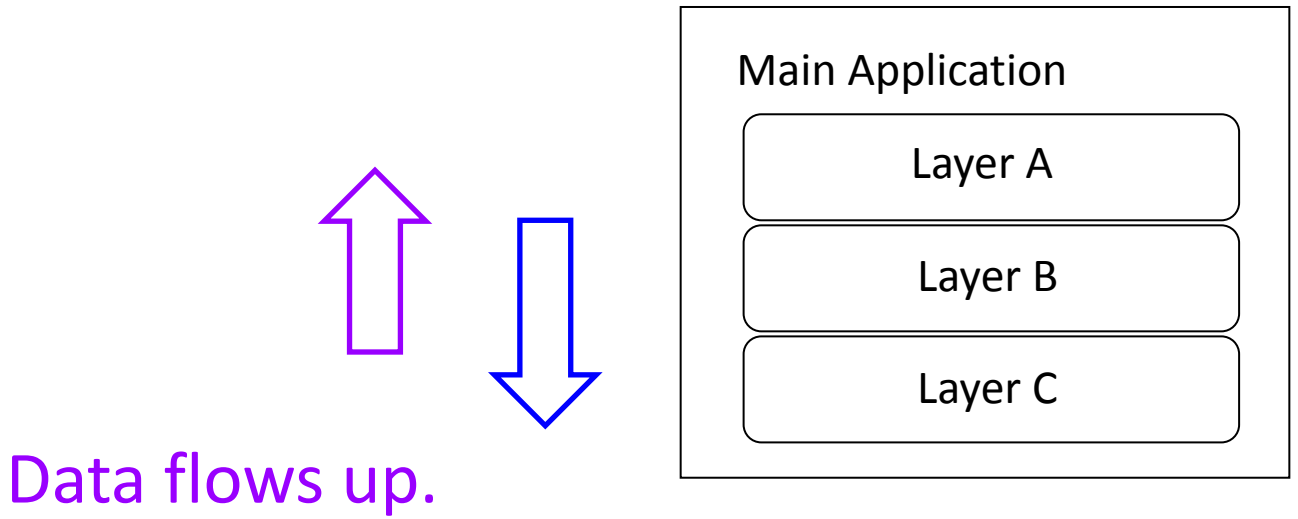
- Layered view
  - Shows ***logical*** organization, ordering based on usage.

- Tiered view
  - Shows ***deployment organization***.

Often combined in a single view.

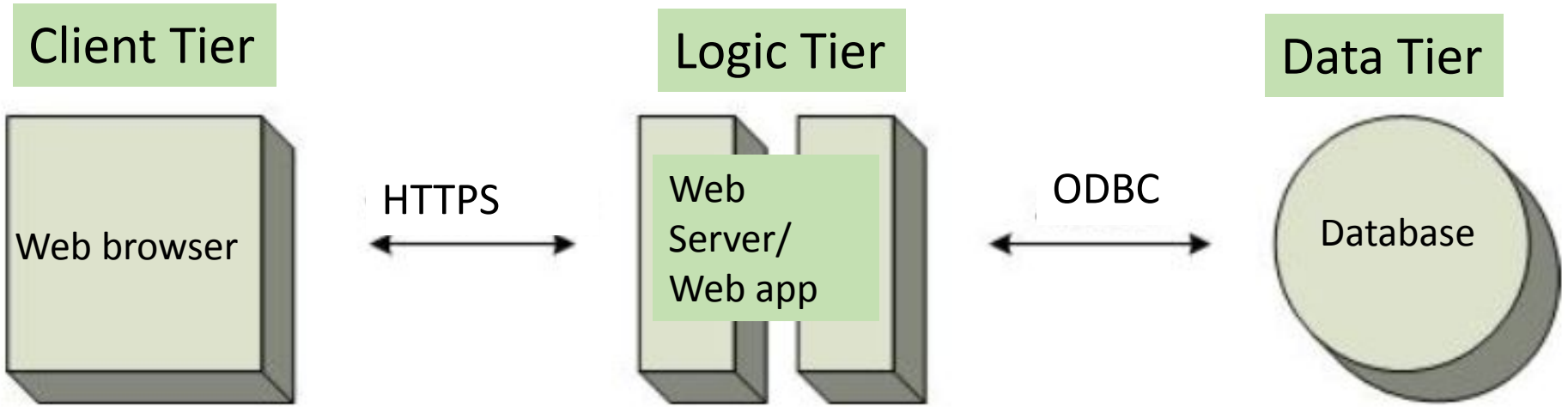
Layer Interaction

- Layer uses the next layer down.
- Layer A uses layer B, but not layer C.
- Layer B does not use layer A.

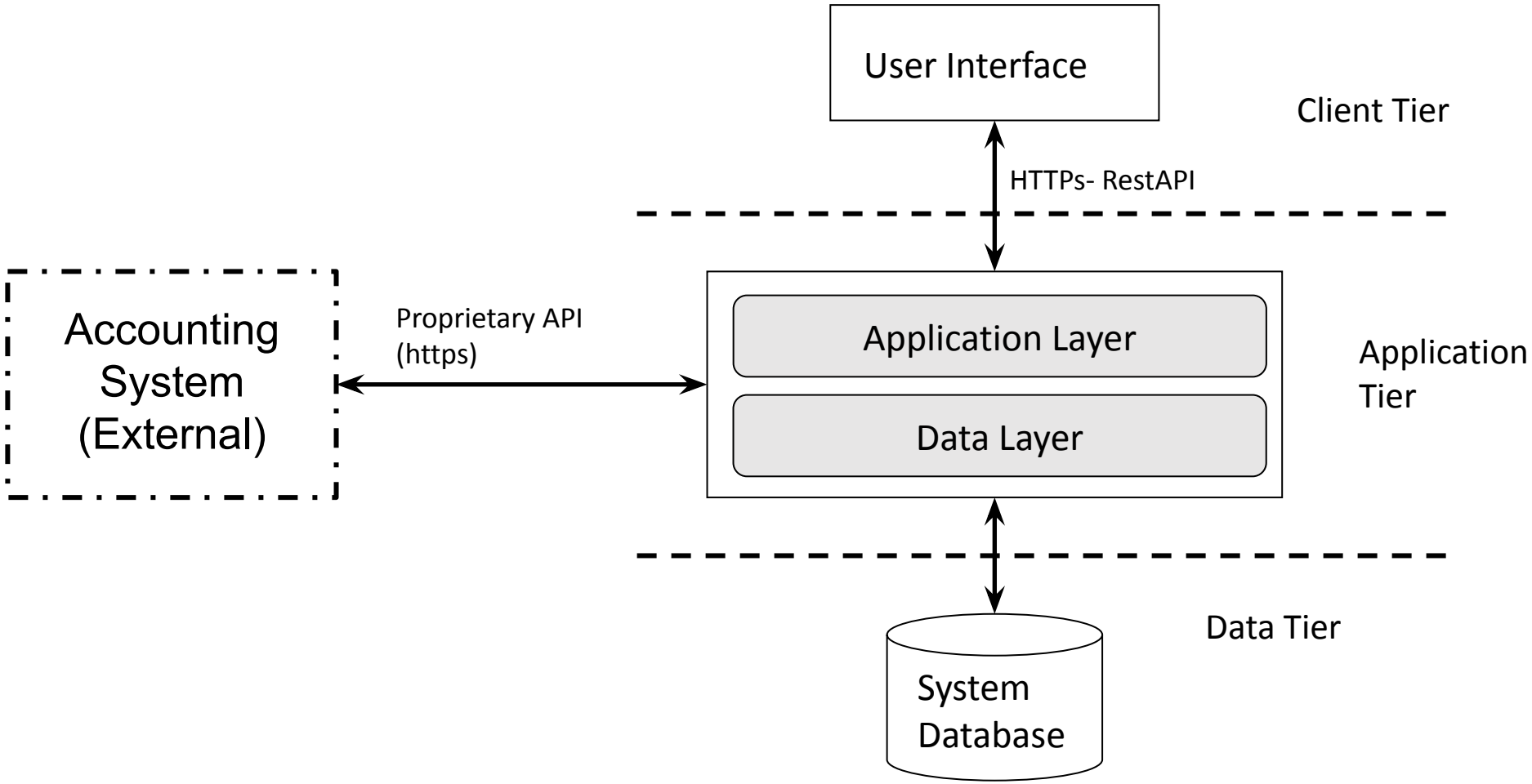


Tiered View of a Full-Stack

- A "Tier" is a unit of **deployment**.

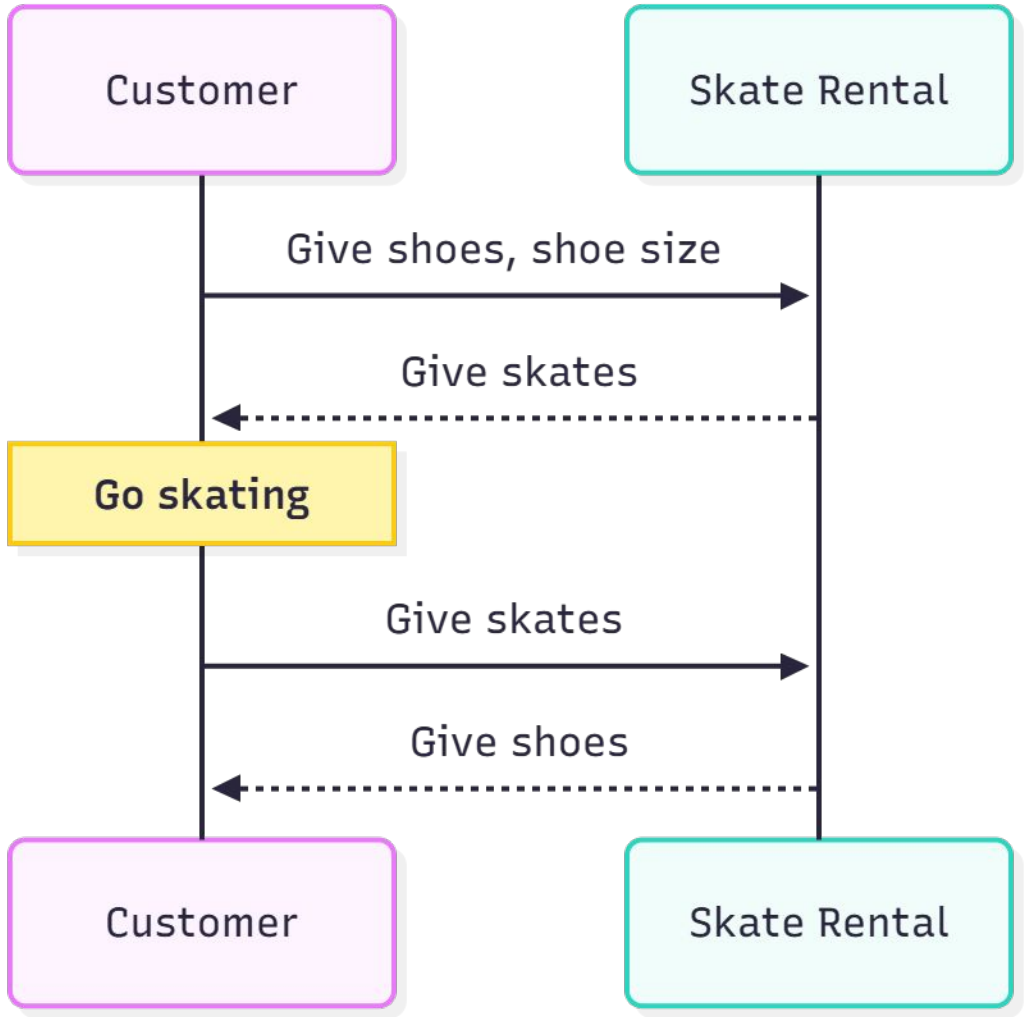


Combined Layer/Tier Architecture view



Interface vs. Implementation

What does the customer need to know to rent skates?



Separation of Responsibilities

Skate rental: Where do we handle payment?

- Option 1: Cashier & skate rental combined
- Option 2: Separate payment & skate rental counters

What are the tradeoffs of these options?

Coupling

Kitchen stove interface:

- Turn knob up to increase temperature
- Turn knob down to increase temperature
- Stove has fixed number of burners

Does using the stove depend on any implementation details?

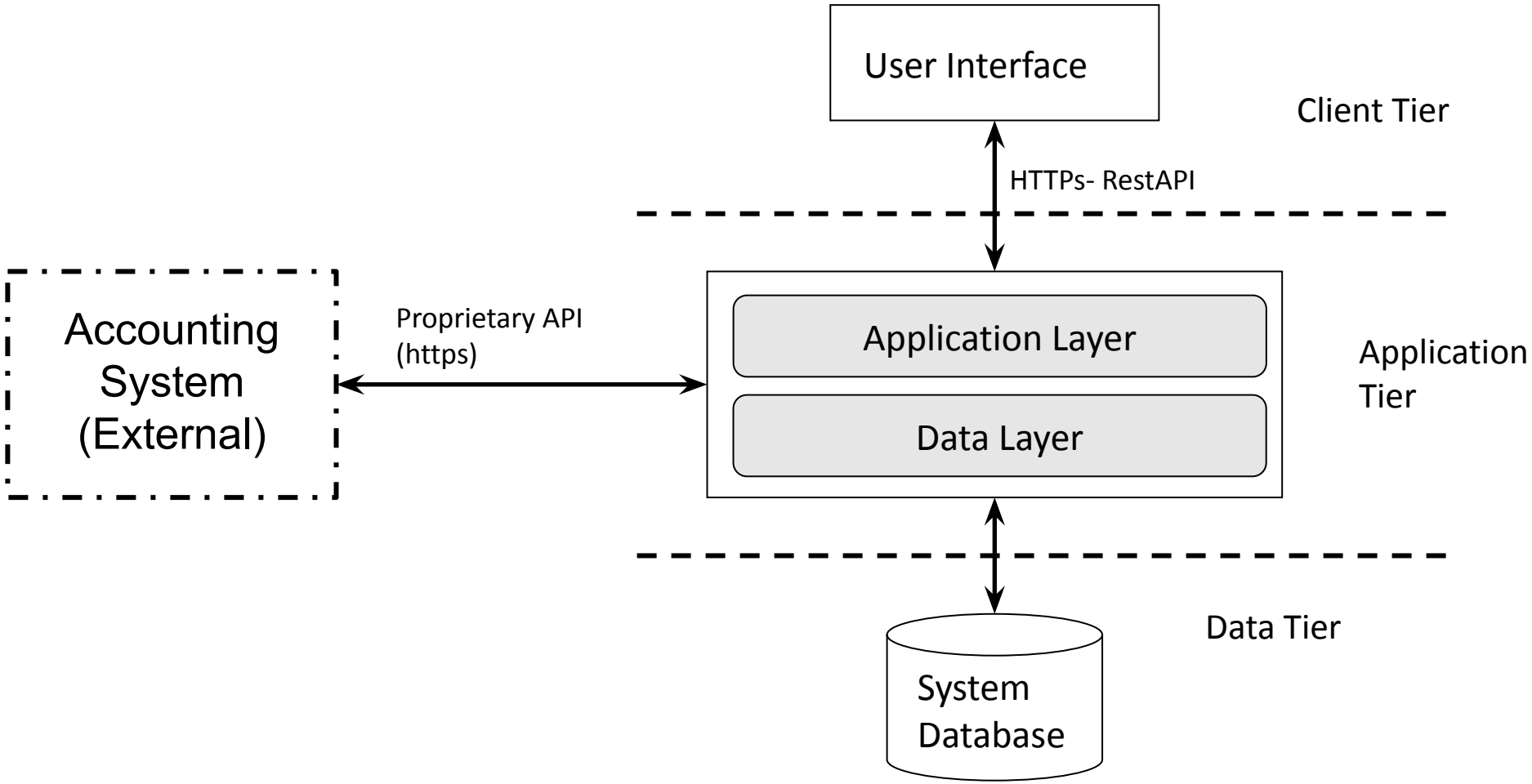
Tightly- vs. Loosely- Coupled

Stove example: **Tightly-coupled**

Skating rink example: **Loosely-coupled**

What are barriers to designing loosely-coupled components?

Why do these things matter?



Conclusion

- We discussed examples of:
  - Separation of **Interface** from **Implementation**
  - **Tight Coupling** vs **Loose Coupling**
  - Division of **responsibilities** across components

- A**rchitecture view diagrams** communicate how **components** of a system are **connected**
  - **Layered** views show how **logical** parts interact
  - **Tiered** views show units of **deployment**