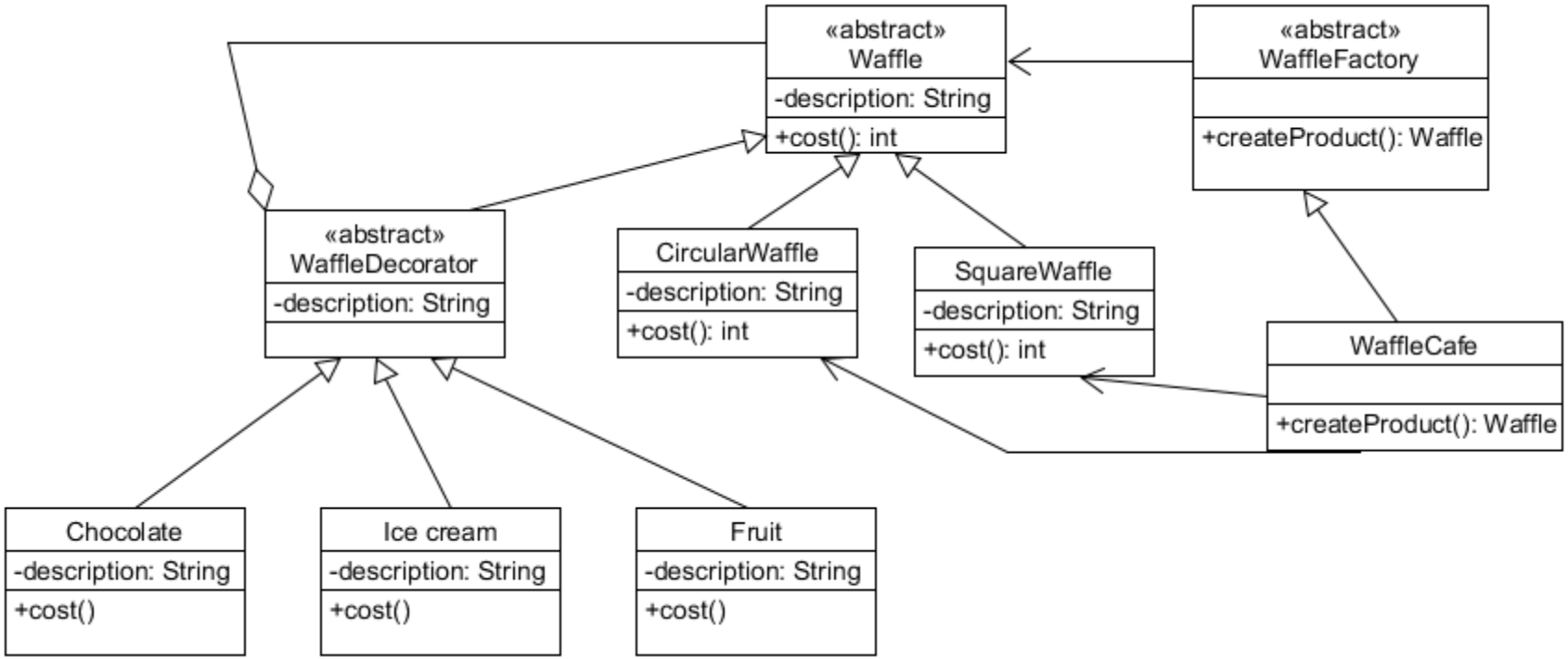
Design Examples
More on Software Architecture

Design Example 1

- Consider a café that where customers can create a waffle. There are two kinds of waffles such as Circular and Square in the café. Several additions can be added to the waffles for extra cost including chocolate, fruit, and ice cream.



Design Example 2

- Consider a graphics library that allows creating images from those types: GIF, JPEG, and BMP. Once an image is created, the library allows users to display the created images in three layouts: stretched layout, squeezed layout, and image-with-filters layout.

More on Software Architecture

System Design Concepts

- Subsystem decomposition
- Services and subsystem interfaces
- Coupling and Coherence
- Layers and partitions

Layers and Partitions

- A layer is a subsystem that provides a service to another subsystem with the following restrictions:
  - A layer only depends on services from lower layers
  - A layer has no knowledge of higher layers
  - The layer that does not depend on any other layer is called the bottom layer, and the layer that is not used by any other is called the top layer

Layers and Partitions

- A layer can be divided horizontally into several independent subsystems called partitions
  - Partitions provide services to other partitions on the same layer
  - Partitions are also called “weakly coupled” subsystems.

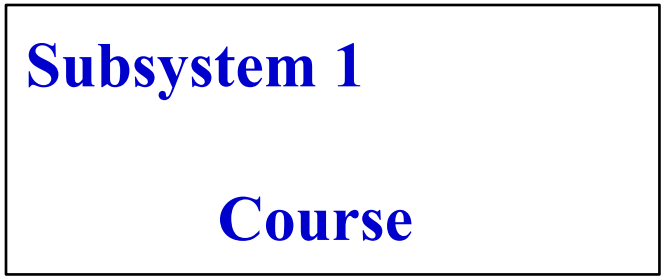
Layers and Partitions

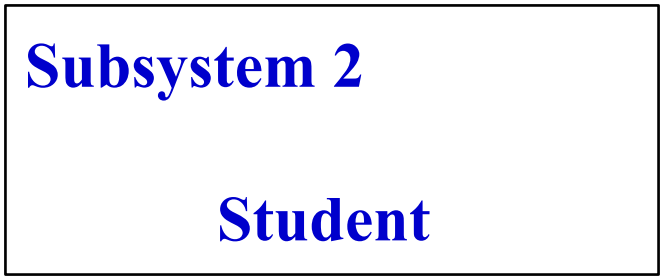
- Two major types of Layer relationships
  - Layer A "depends on" Layer B (compile time dependency)
    - Example: Build dependencies (make, ant, maven)
  - Layer A "calls" Layer B (runtime dependency)
    - Example: A web browser calls a web server
    - Can the client and server layers run on the same machine?
      - Yes, they are layers, not processor nodes
      - Mapping of layers to processors is decided during the Software/hardware mapping!

Layers and Partitions

- Partition relationship
  - The subsystems have mutual knowledge about each other
    - A calls services in B; B calls services in A (Peer-to-Peer)
- UML convention:
  - Runtime dependencies are associations with dashed lines
  - Compile time dependencies are associations with solid lines.

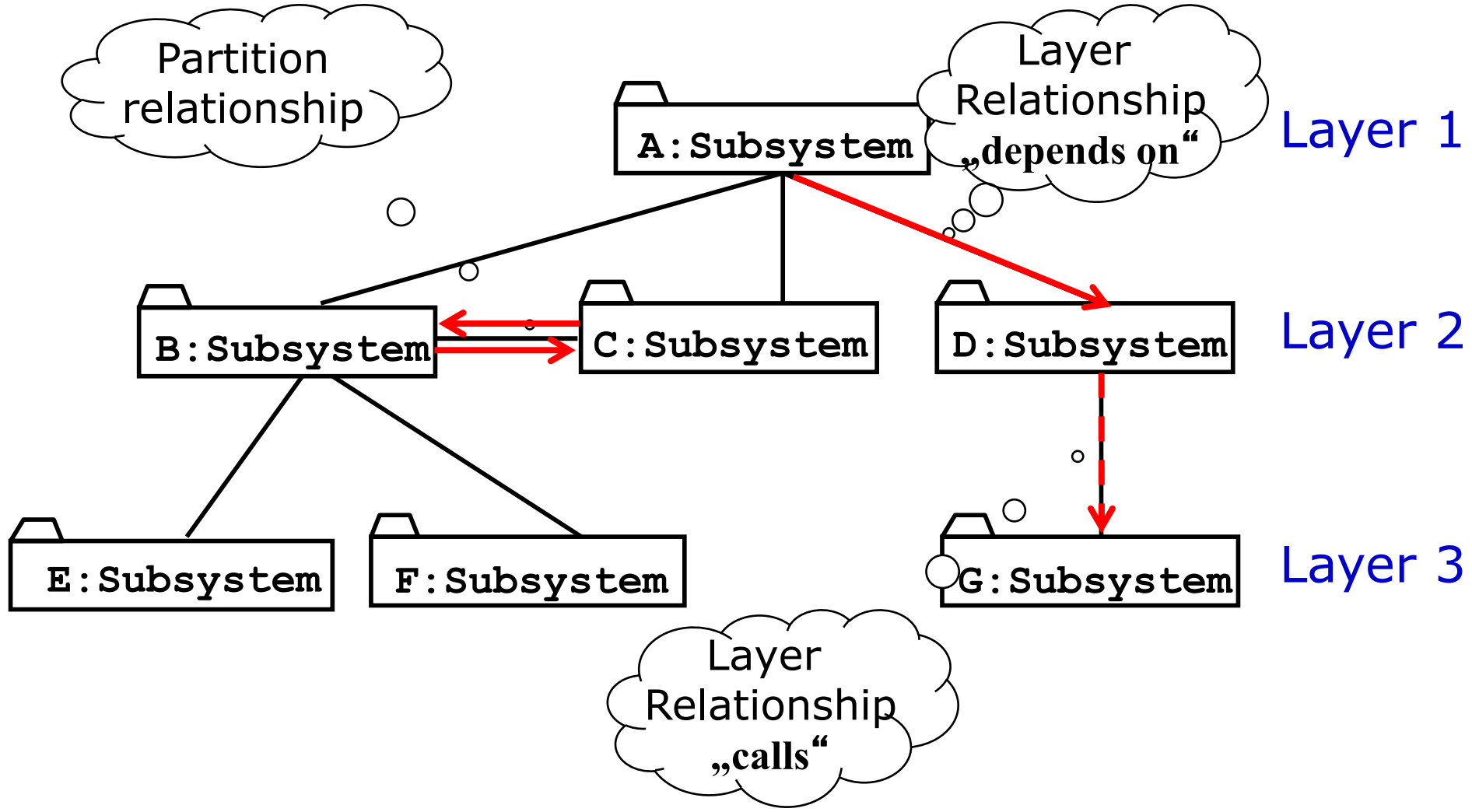
```
Class Course {
    code, name, duration,…
}
```






```
Class Student{
    name, age, Course[6] registeredCourses;
}
```

Example of a Subsystem Decomposition into Layers

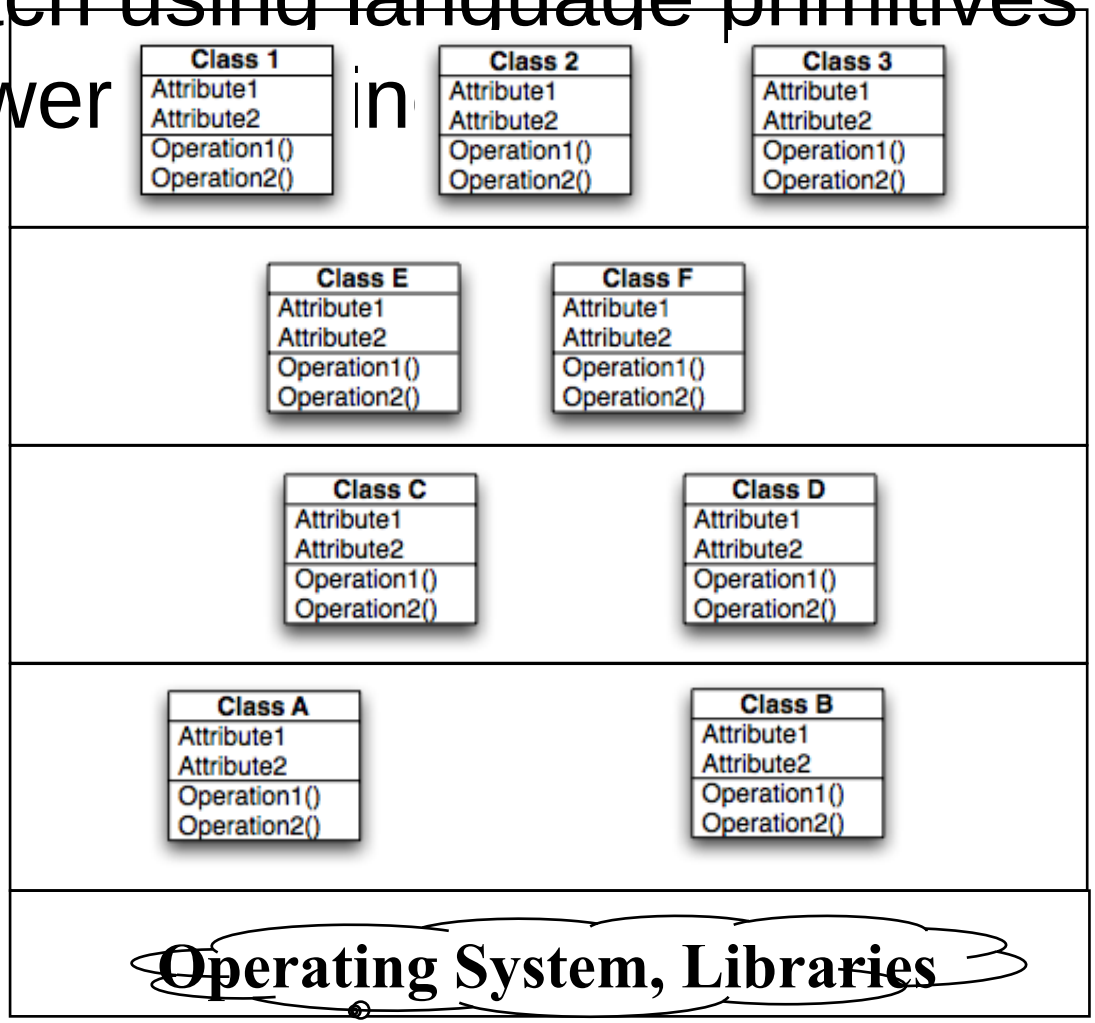


Virtual Machine

- A virtual machine is a subsystem connected to higher and lower level virtual machines by "provides services for" associations
- A virtual machine is an abstraction that provides a set of attributes and operations
- The terms layer and virtual machine can be used interchangeably
  - Also sometimes called “level of abstraction”.

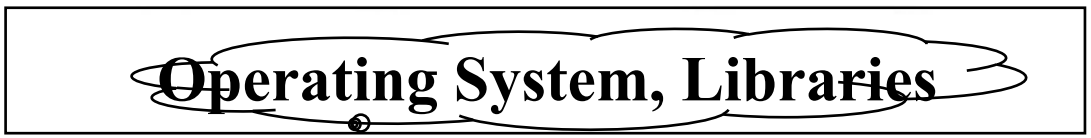
Building Systems as a Set of Virtual Machines

A system is a hierarchy of virtual machines, each using language primitives offered by the lower in



Building Systems as a Set of Virtual Machines

A system is a hierarchy of virtual machines,

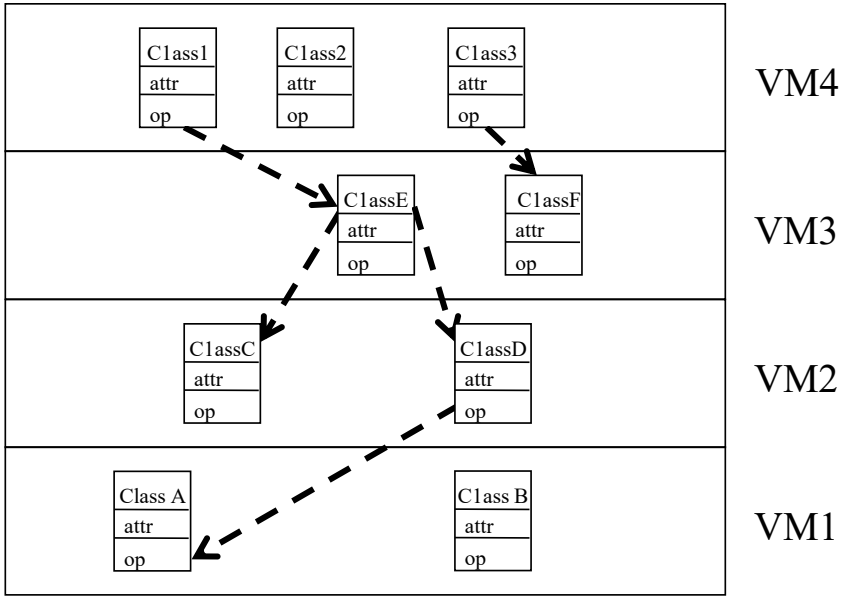


Existing System

Option 1: Closed Architecture (Opaque Layering)

- Each virtual machine can only call operations from the layer below

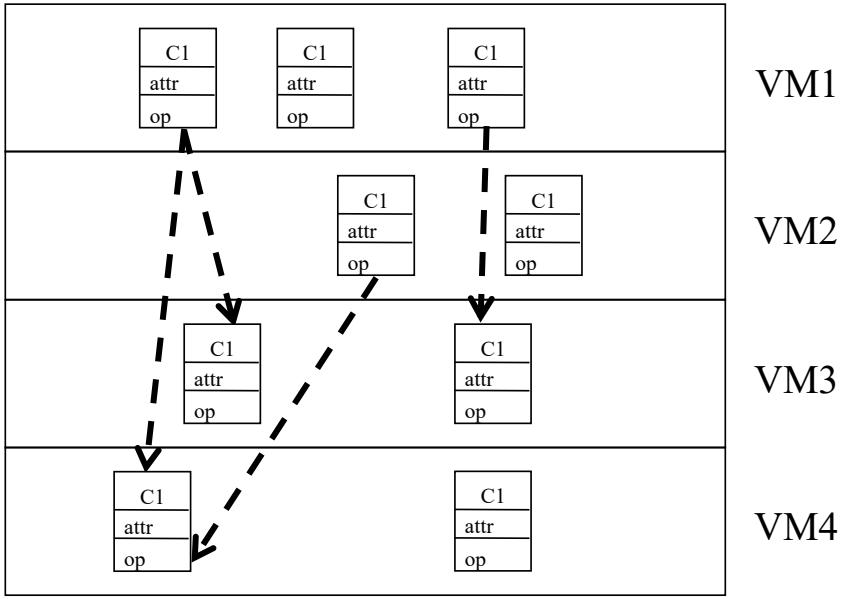
Design goals:
Maintainability.



Option 2: Open Architecture (Transparent Layering)

- Each virtual machine can call operations from any layer below

Design goal:
Runtime efficiency



Required Reading

- Chapter 6 from": Bruegge's and Dutoit's "Object-Oriented Software Engineering", Prentice Hall, Third Edition, 2010.