Phân tích thiết kế với UML

Bài 5: Kiến trúc hệ thống

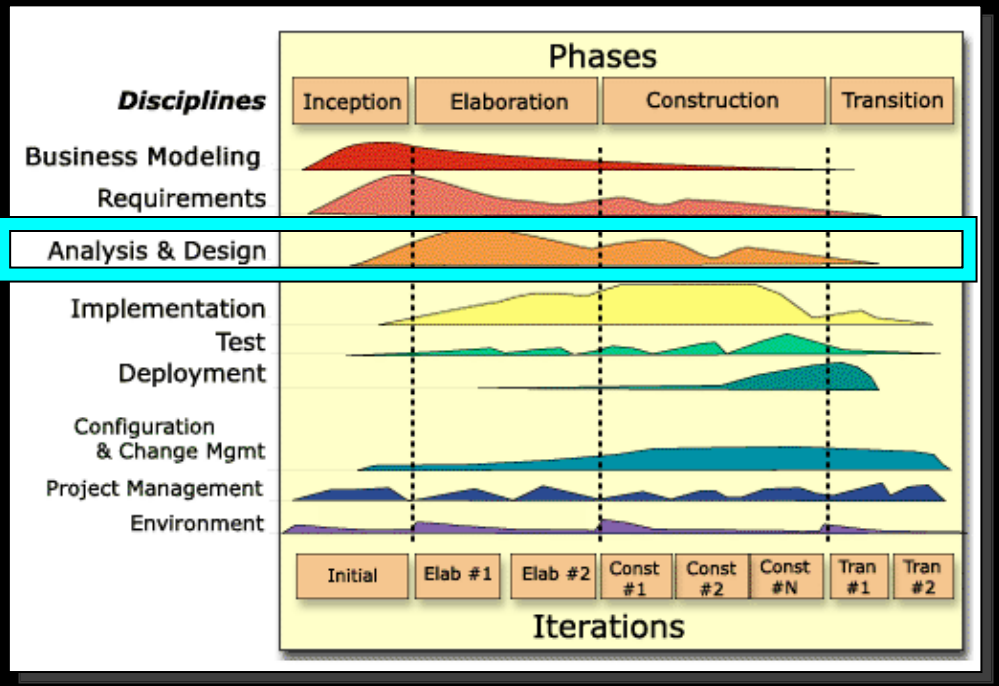
Nhắc lại kiến thức

- Mục đích của Phân tích và Thiết kế?
- Đầu vào và Đầu ra?
- Sự khác biệt giữa Phân tích và Thiết kế?
- Tiếp cận usecase-driven trong PT&TK
- Các góc nhìn kiến trúc – mô hình 4+1

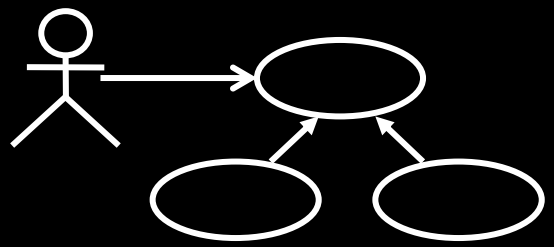
Mục đích của PT&TK

Mục đích của phân tích và thiết kế:

- **Chuyển yêu cầu phần mềm thành một bản thiết kế.**
- **Đúc rút một bản kiến trúc tốt.**
- **Làm cho bản thiết kế thích ứng với môi trường triển khai và có hiệu năng tốt nhất.**

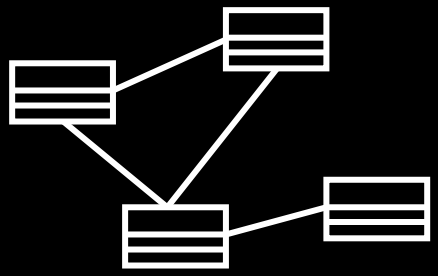


Đầu vào và đầu ra của PT&TK

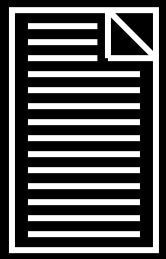


Mô hình Ca sử dụng

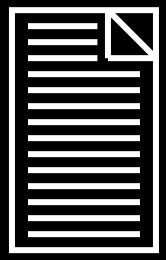
Phân tích và Thiết kế



Mô hình thiết kế

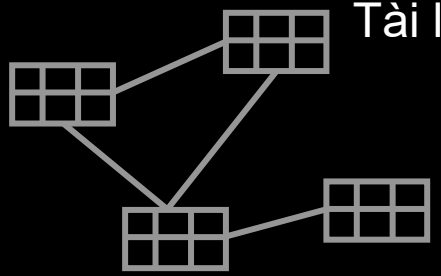


Bảng chú giải



Đặc tả bổ sung

Tài liệu kiến trúc

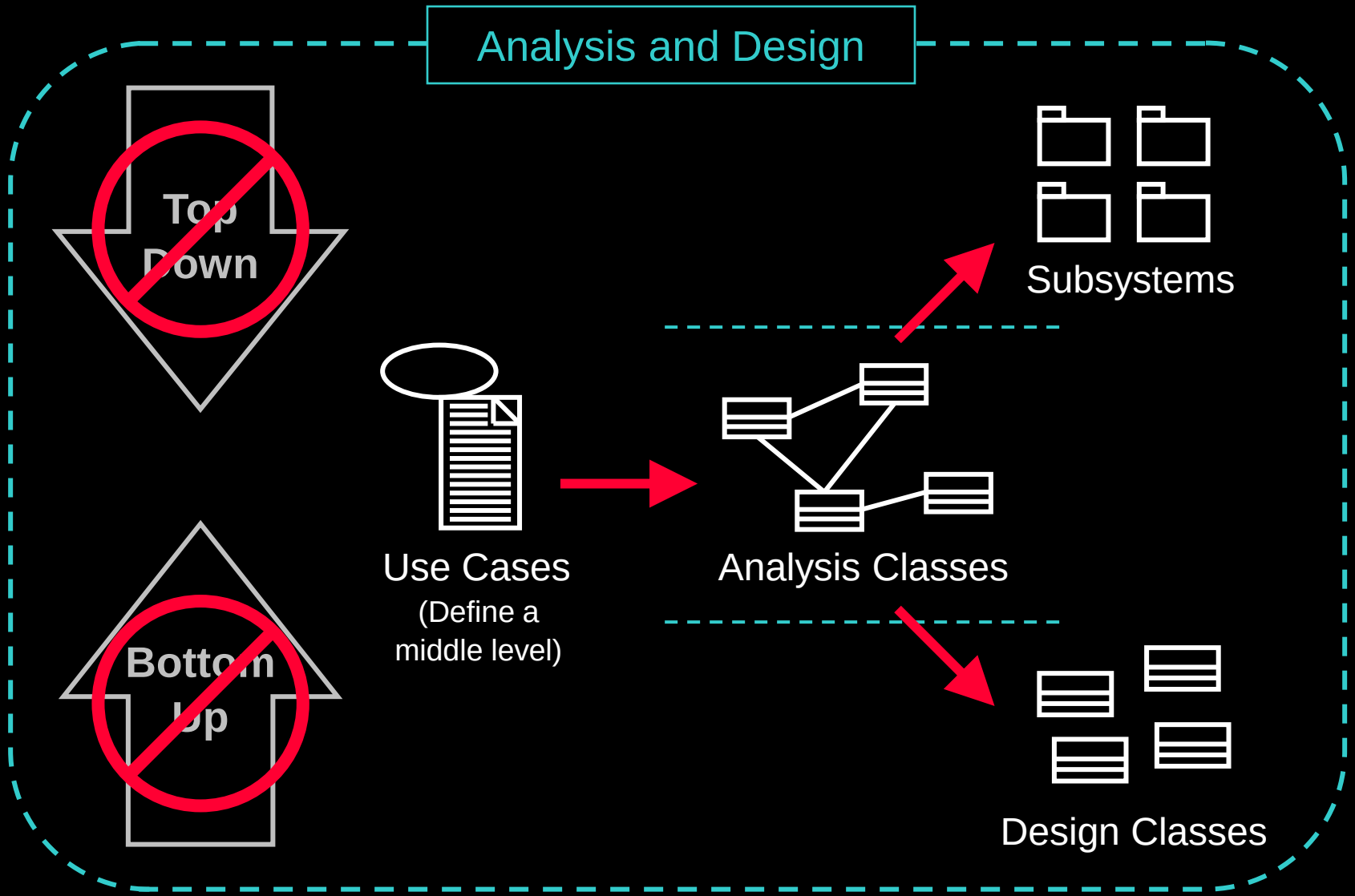


Mô hình dữ liệu

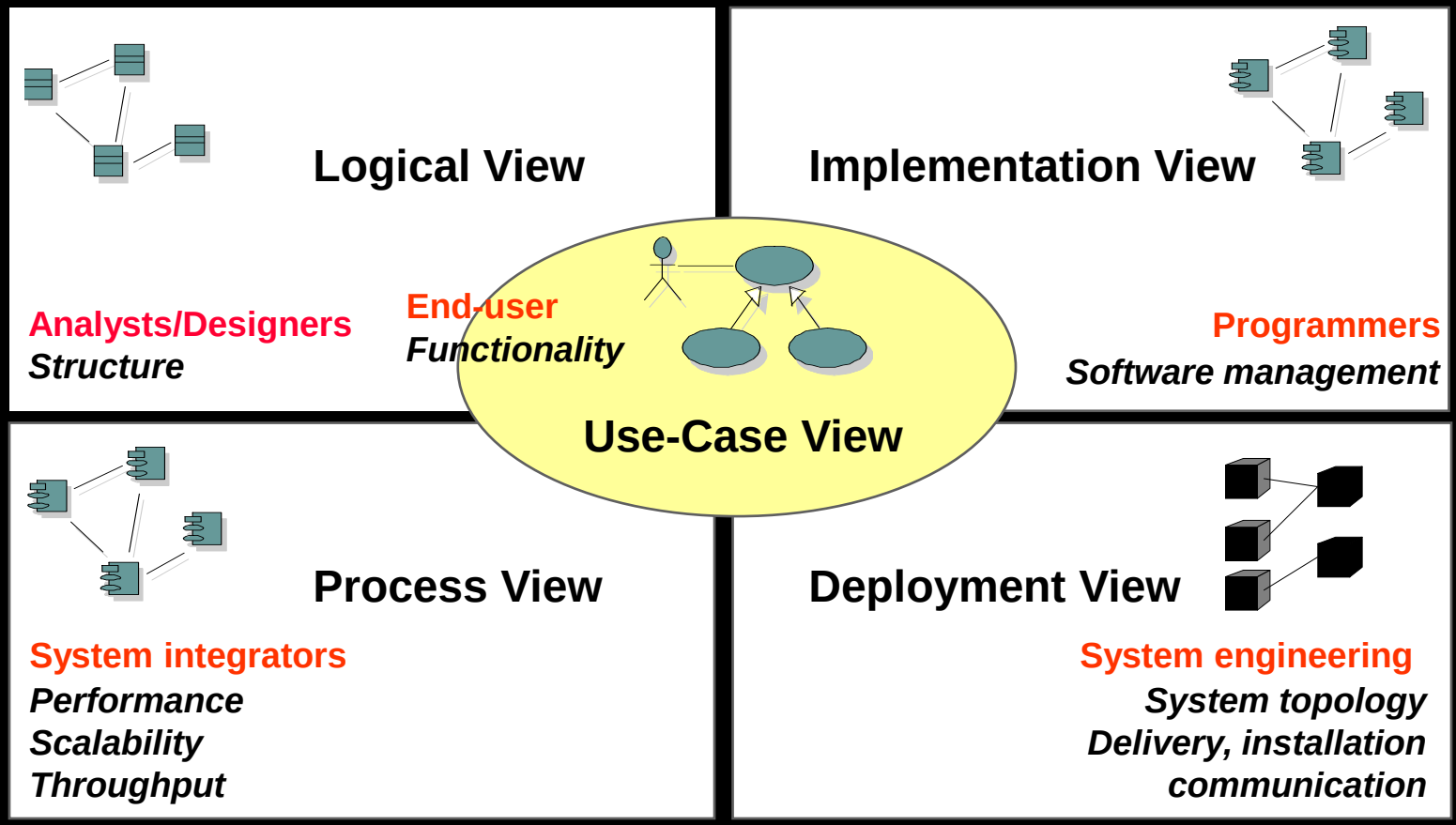
Phân biệt Phân tích với Thiết kế

- Phân tích
  - Tập trung vào việc tìm hiểu vấn đề
  - Bản thiết kế lý tưởng
  - Nắm bắt hành vi
  - Nắm bắt cấu trúc hệ thống
  - Đáp ứng các yêu cầu chức năng
  - Mô hình cỡ nhỏ
- Thiết kế
  - Tập trung vào việc tìm hiểu giải pháp
  - Xác định thao tác và thuộc tính
  - Đảm bảo hiệu năng
  - Xác định vòng đời đối tượng
  - Đáp ứng các yêu cầu phi chức năng
  - Mô hình cỡ lớn

Tiếp cận Dẫn dắt bởi ca sử dụng (usecase-driven)



Kiến trúc phần mềm: Mô hình “4+1 View”



Nội dung

- Kiến trúc là gì?
- Giải thích vai trò của kiến trúc trong vòng đời phát triển phần mềm.
- Giới thiệu khái niệm:
  - Mẫu kiến trúc – architectural pattern
  - Kỹ thuật - mechanism

Kiến trúc phần mềm

- Kiến trúc phần mềm bao gồm một tập các quyết định về tổ chức của phần mềm.
  - Sự lựa chọn các yếu tố cấu trúc và giao diện của chúng để lắp ghép thành hệ thống
  - Sự cộng tác của các yếu tố cấu trúc
  - Sự lắp ghép, tích hợp các yếu tố cấu trúc và hành vi để tạo nên các hệ thống con
  - Phong cách kiến trúc

Grady Booch, Philippe Kruchten, Rich Reitman, Kurt Bittner; Rational
(derived from Mary Shaw)

Kiến trúc phần mềm

- Kiến trúc phần mềm cung cấp các góc nhìn khác nhau về một hệ thống phần mềm.
  - Góc nhìn logic:
    - Hệ thống được cấu thành từ đối tượng thuộc những lớp gì?
    - Sự cộng tác của các đối tượng này như thế nào?
  - Góc nhìn thực hiện
    - Các lớp thuộc hệ thống được xây dựng từ những thành phần mã nguồn gì?
    - Chúng được biên dịch/đóng gói thành các thành phần thực thi/thành phần bố trí gì?

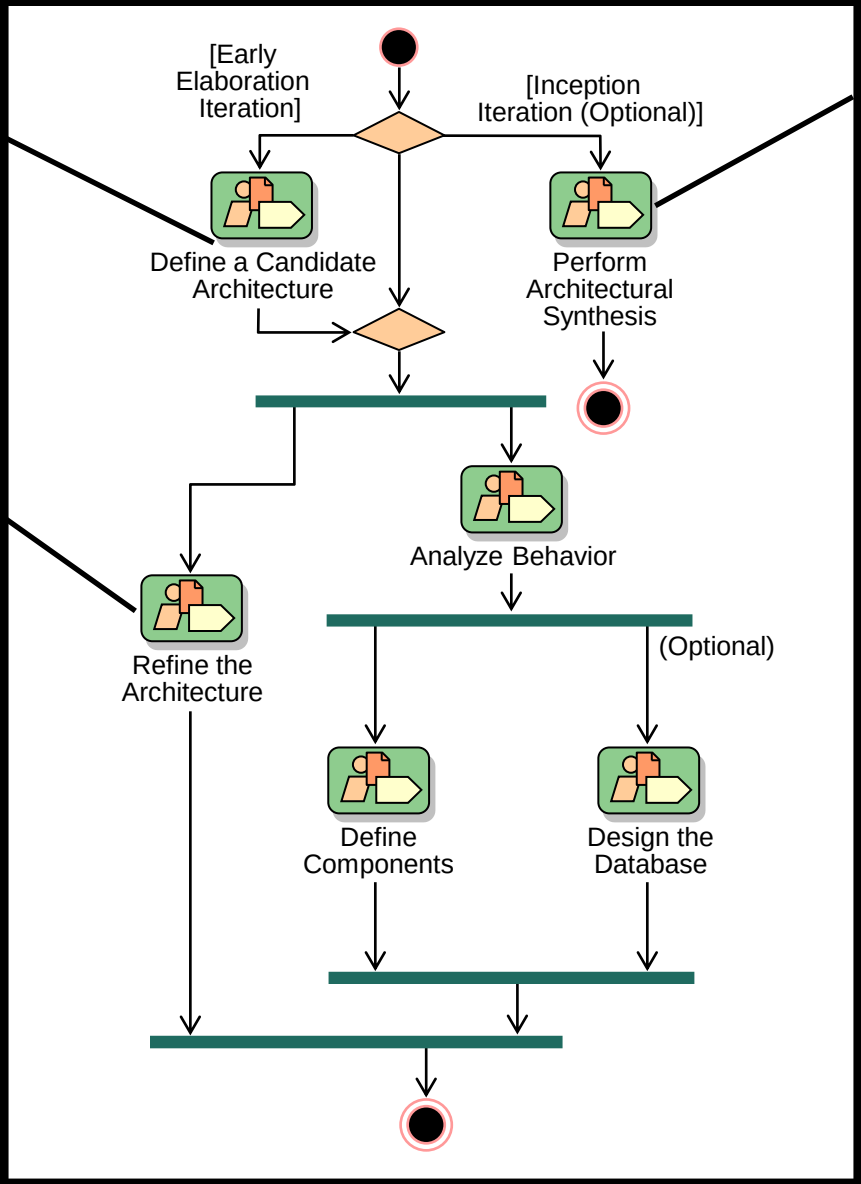
Kiến trúc phần mềm

- Kiến trúc phần mềm cung cấp các góc nhìn khác nhau về một hệ thống phần mềm.
  - Góc nhìn tiến trình:
    - Hệ thống được thực hiện trên các luồng/tiến trình gì?
    - Đối tượng thuộc những lớp nào hoạt động trên từng luồng/tiến trình
  - Góc nhìn triển khai
    - Hệ thống phần mềm được triển khai trên hạ tầng phần cứng thế nào?
    - Các thành phần bố trí được đặt trên các nút tính toán như thế nào?

Kiến trúc trong vòng đời phát triển phần mềm

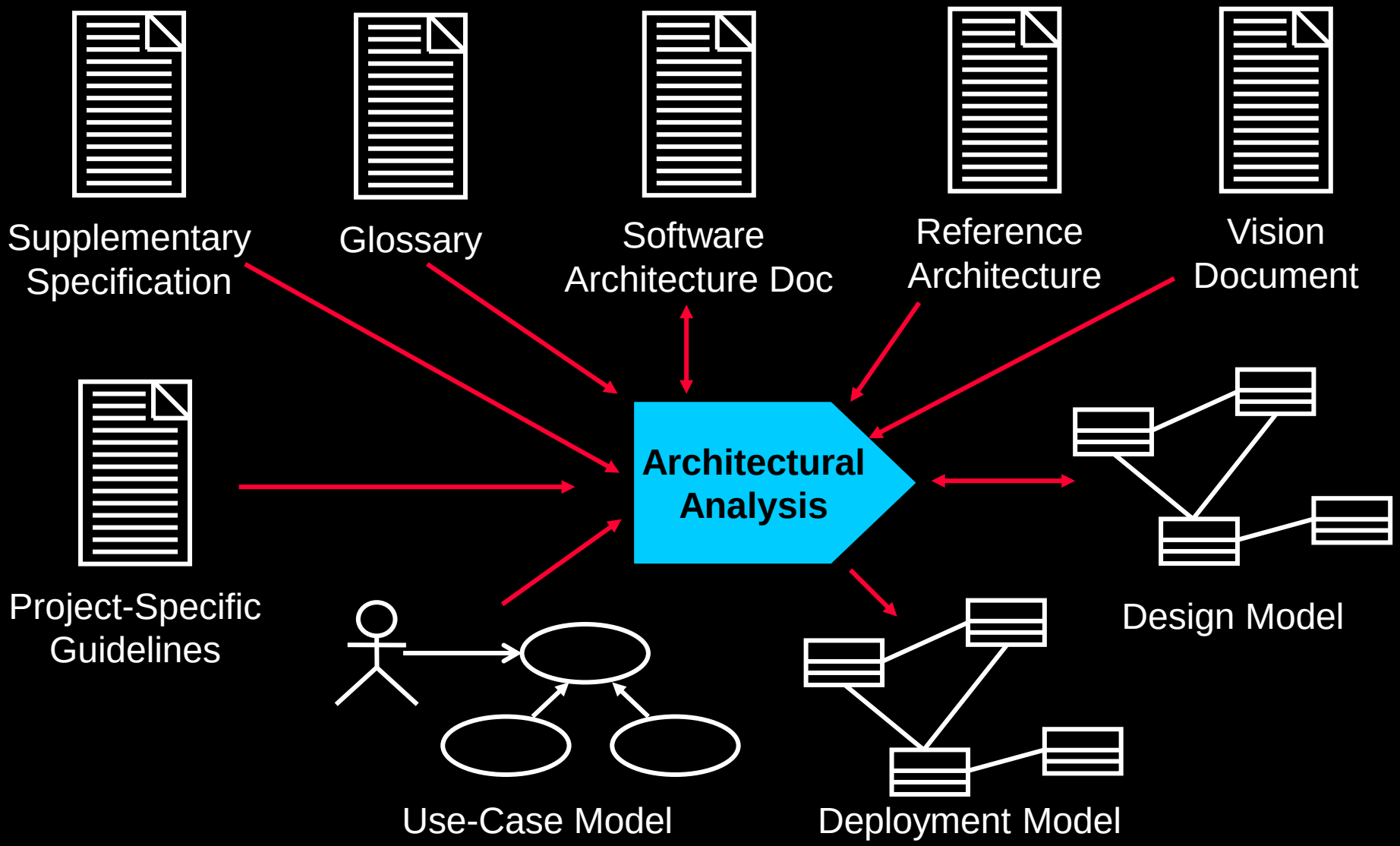
Trong bước lặp đầu tiên của pha Elaboration, kiến trúc sư xác định một khung kiến trúc cho hệ thống (từ việc tổng hợp các mẫu kiến trúc sẵn có) Bước này thường được gọi là: **Phân tích kiến trúc**

Trong trình phân tích thiết kế, khung kiến trúc sẽ được bổ sung, mở rộng với các yếu tố thiết kế để tạo thành bản thiết kế kiến trúc hoàn chỉnh



Trong pha đầu, kiến trúc sư thu thập, tổng hợp những mẫu thiết kế kiến trúc sẵn có

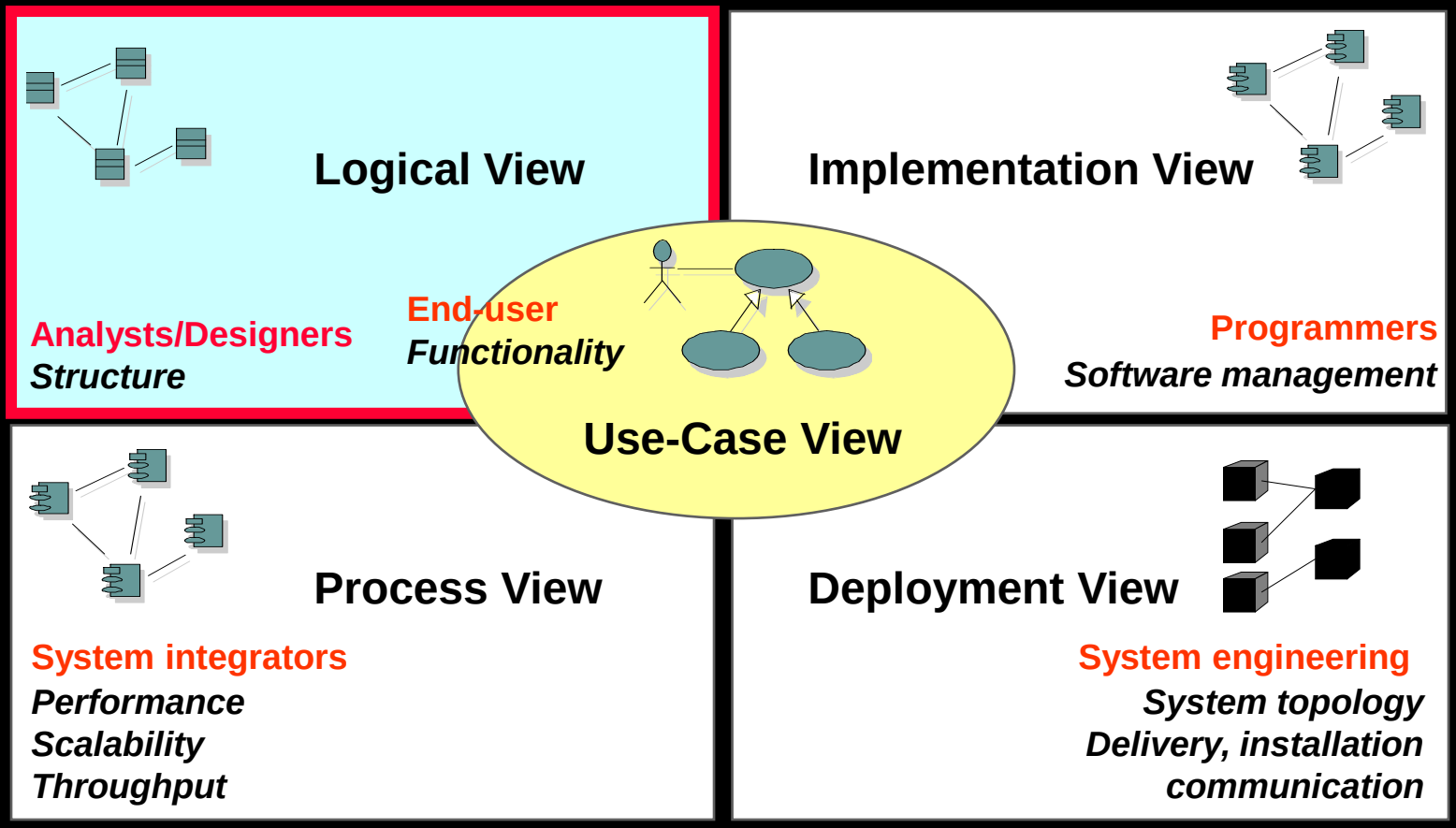
Phân tích kiến trúc



Tổ chức mức cao của hệ thống

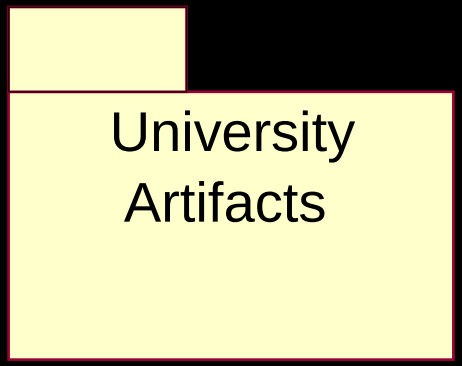
- Để có được góc nhìn toàn cảnh của hệ thống
  - Hệ thống được cấu thành từ các hệ thống con nào
  - Các hệ thống con có mối quan hệ với nhau như thế nào
- Tổ chức mức cao thường được trình bày dưới dạng biểu đồ gói dưới khung nhìn logic

“4+1 View” Model

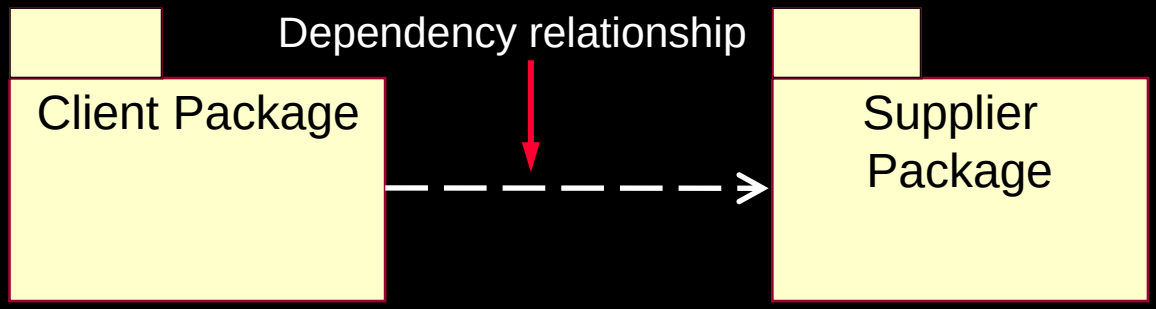


Ôn lại khái niệm gói

- Gói là kỹ thuật gom nhiều yếu tố liên quan lại thành nhóm.
- Gói có thể chứa các gói nhỏ hơn.

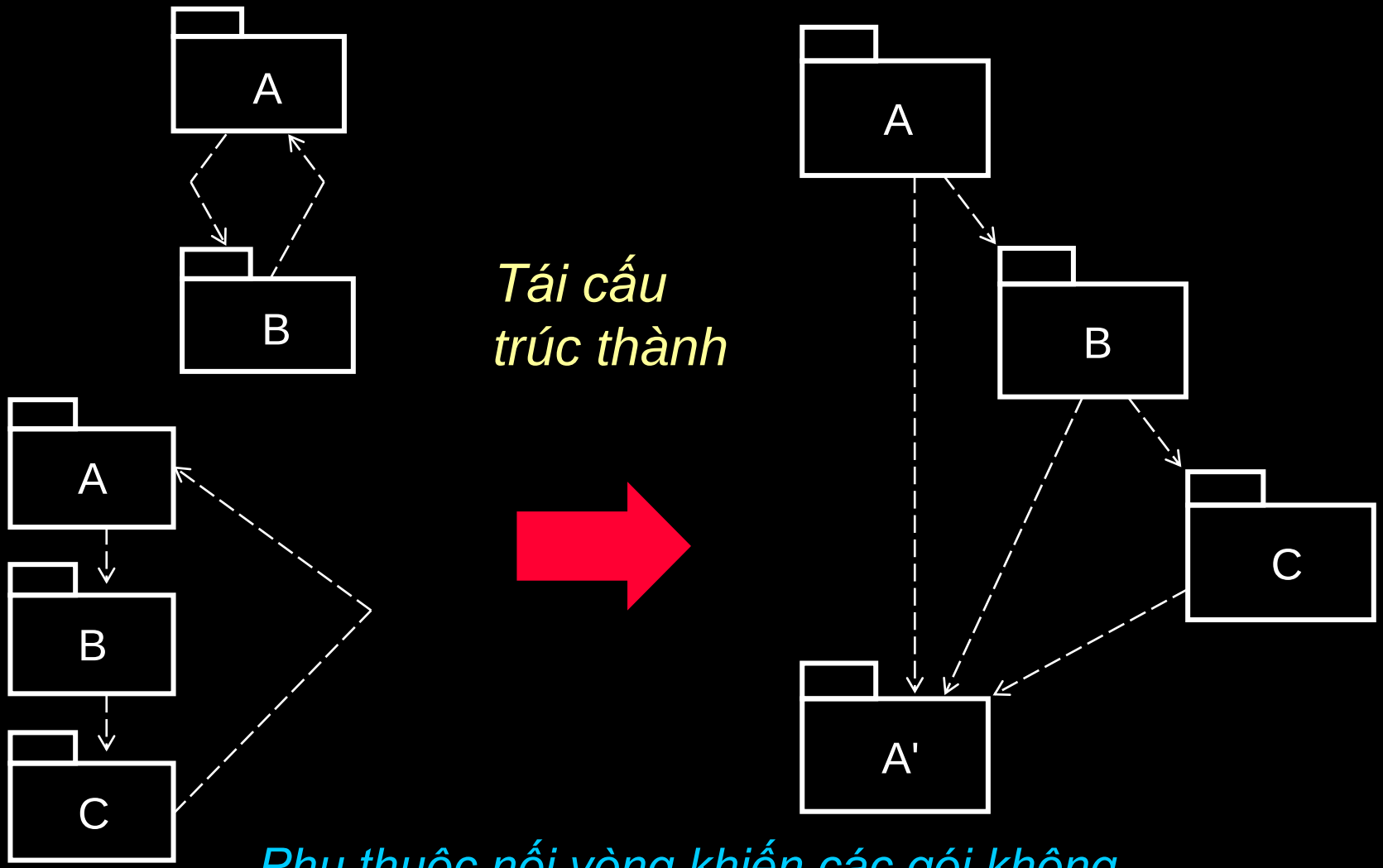


Quan hệ phụ thuộc giữa các gói



- Quan hệ phụ thuộc
  - Sự thay đổi của gói Supplier có thể ảnh hưởng tới gói Client.
  - Gói Client không thể tái sử dụng một cách độc lập bởi nó phụ thuộc vào gói Supplier.

Hạn chế phụ thộc nối vòng giữa các gói



Phụ thuộc nối vòng khiến các gói không thể tái sử dụng một cách độc lập

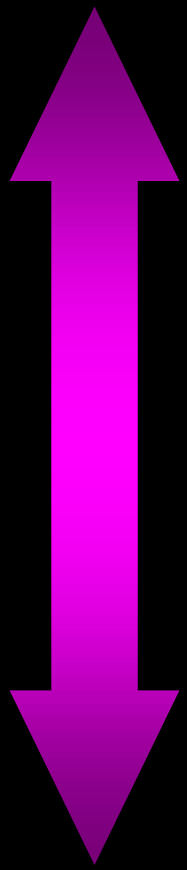
Mẫu kiến trúc

- Tổ chức mức cao của các phần mềm thường có khung giống nhau
- Trong quá trình phân tích, kiến trúc sư thường lựa chọn khung kiến trúc cho hệ thống từ việc tổng hợp từ các kiến trúc sẵn có.
- Mẫu kiến trúc là một kỹ thuật khái quát hóa các kiến trúc có tổ chức tương tự nhau
  - Layers
  - Model-view-controller (M-V-C)
  - Pipes and filters
  - Blackboard

Khái niệm Mẫu và Khung

- Mẫu - Pattern
  - Cung cấp một giải pháp chung cho một lớp vấn đề tương tự
- Khung - Framework
  - Định nghĩa một nền tảng cho các giải pháp, từng giải pháp cụ thể có thể được phát triển, mở rộng từ nền tảng này

Mẫu kiến trúc phân tầng - Layering



Application Subsystems

Distinct application subsystems that make up an application — contains the value adding software developed by the organization.

Business-Specific

Business specific — contains a number of reusable subsystems specific to the type of business.

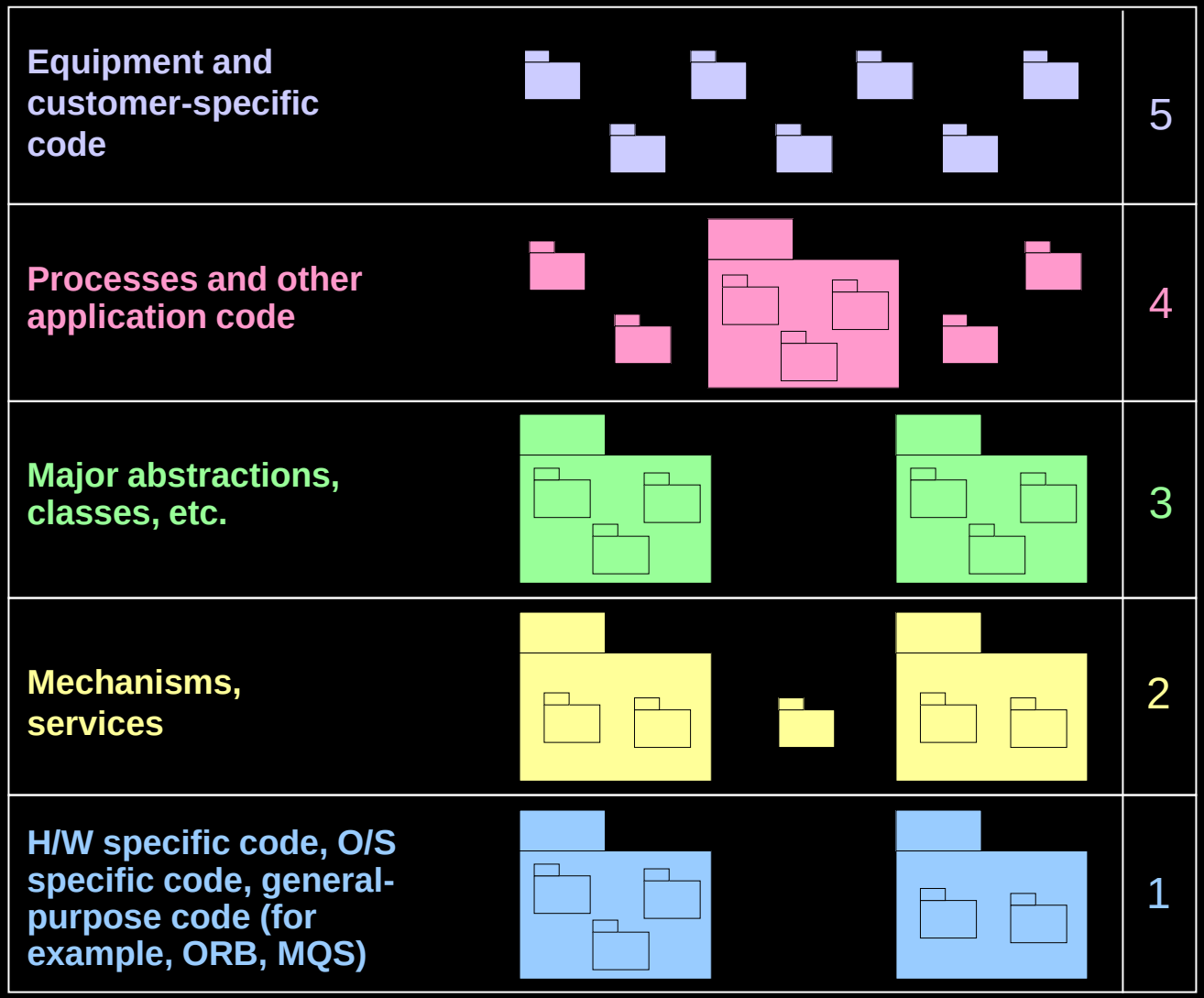
Middleware

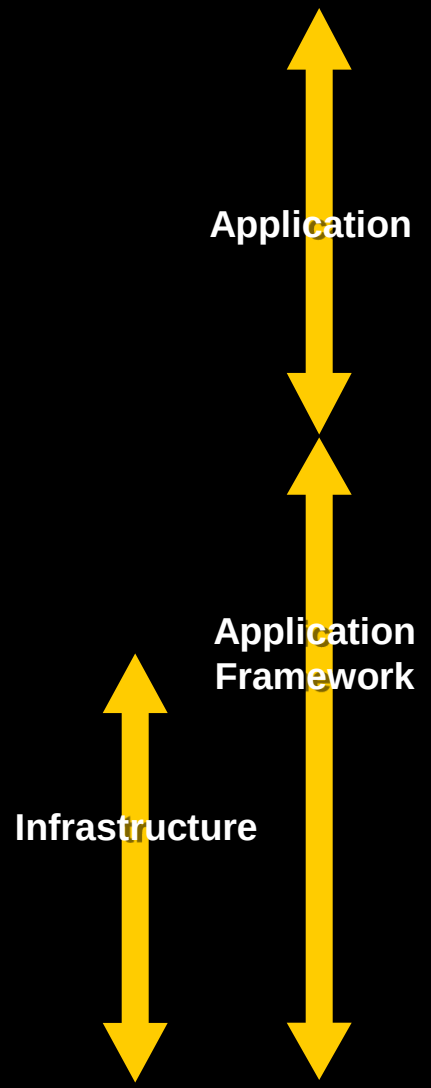
Middleware — offers subsystems for utility classes and platform-independent services for distributed object computing in heterogeneous environments and so on.

System Software

System software — contains the software for the actual infrastructure such as operating systems, interfaces to specific hardware, device drivers, and so on.

Mẫu kiến trúc phân tầng - Layered



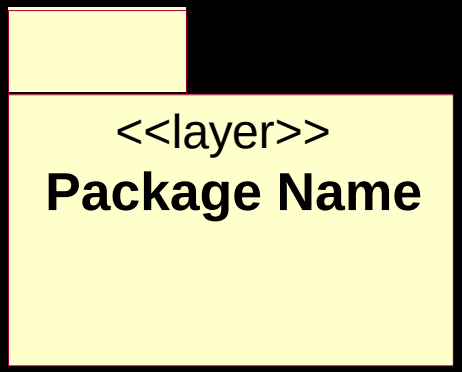


Mẫu kiến trúc phân tầng

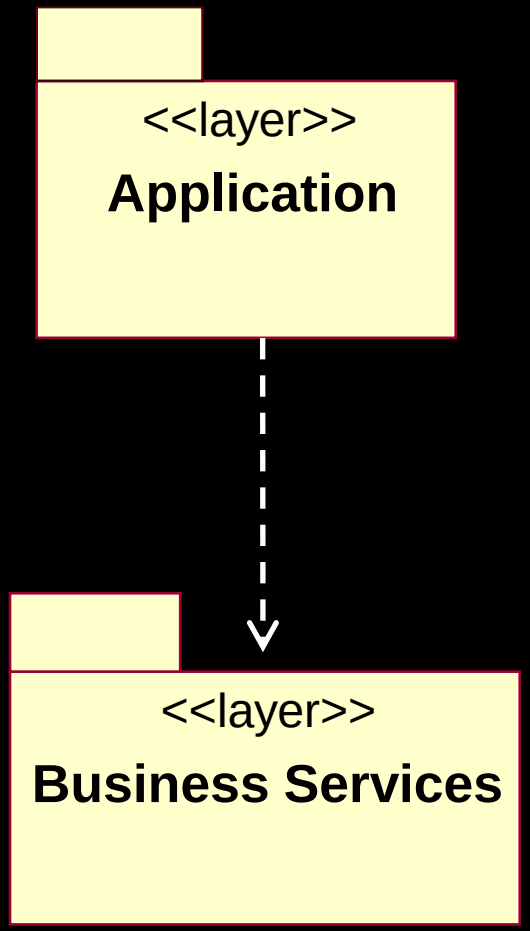
- Phân tách các mức độ trừu tượng
  - Các yếu tố có cùng mức đồ trừu tượng được đặt chung trong một tầng
- Phân tách các mối quan tâm (separation of concerns)
  - Nhóm các yếu tố giống nhau lại
  - Phân tách các yếu tố khác nhau
- Resiliency
  - Ràng buộc lỏng
  - Đóng gói các thay đổi
  - Phân tách các yếu tố thuộc các nhóm khác nhau để có thể dễ thay đổi

Ký pháp UML

- Mỗi tầng có thể được biểu diễn bằng một gói với stereotype <<layer>>



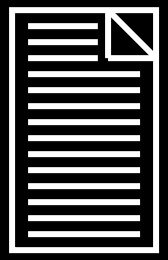
Ví dụ về một tổ chức phân tầng



Kỹ thuật - Mechanisms

Yêu cầu chức năng

Môi trường thực hiện



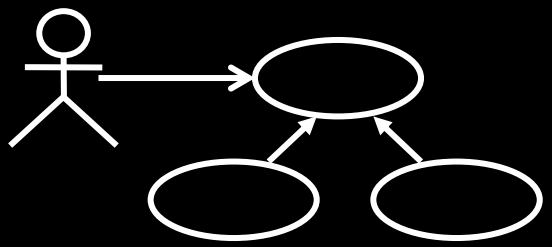
Supplementary Specification

"realized by client classes using"

Mechanisms

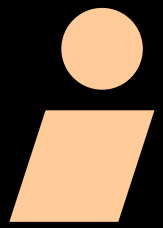
"constrained by"

COTS Products
Databases
IPC Technology
etc.



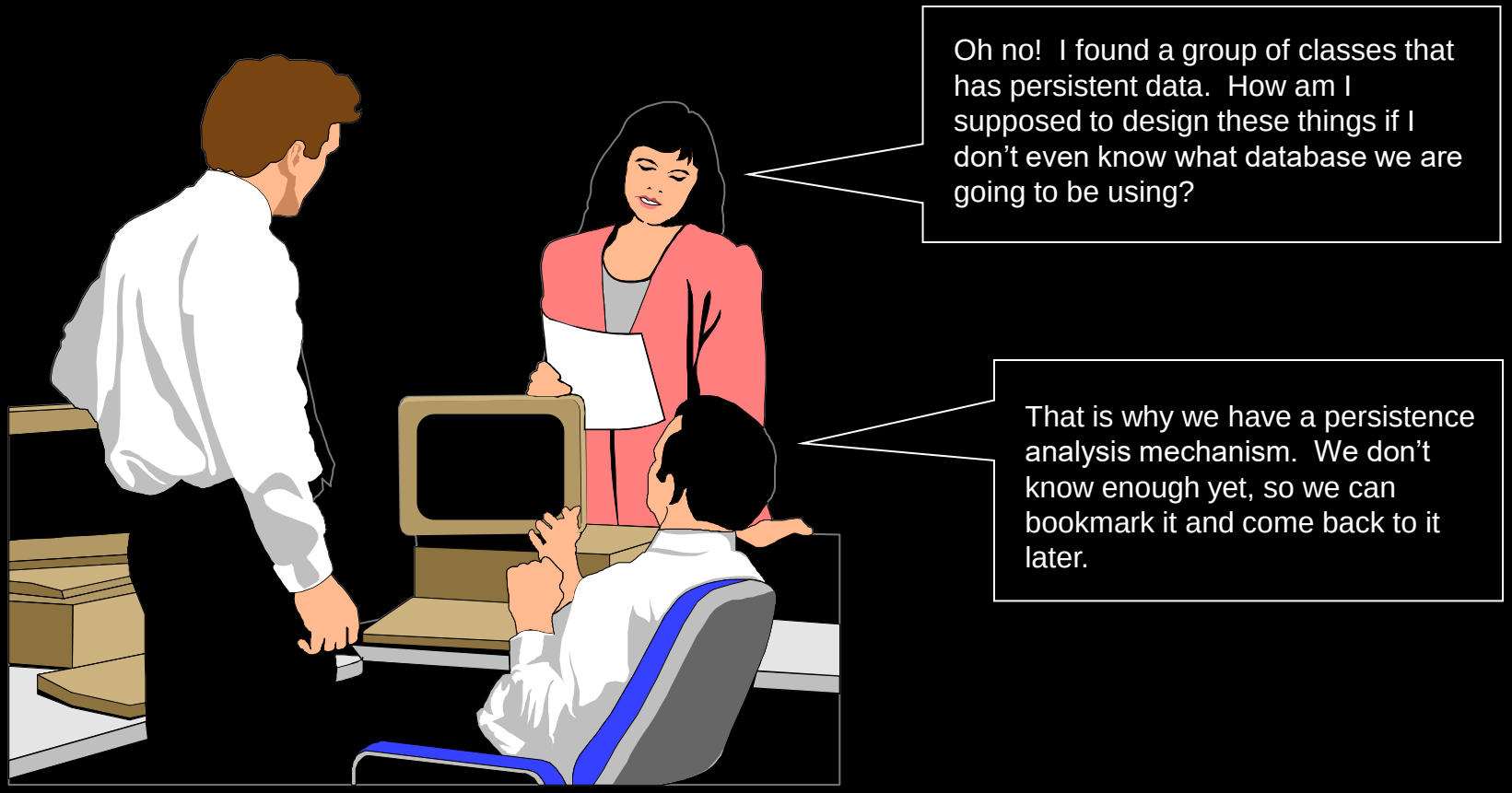
Use-Case Model

"responsible for"



Architect

Kỹ thuật phân tích



Analysis mechanisms are used during analysis to reduce the complexity of analysis and to improve its consistency by providing designers with a shorthand representation for complex behavior.

Ví dụ về các kỹ thuật phân tích

- Persistency
- Communication (IPC and RPC)
- Message routing
- Distribution
- Transaction management
- Process control and synchronization (resource contention)
- Information exchange, format conversion
- Security
- Error detection / handling / reporting
- Redundancy
- Legacy Interface

Đặc tả chi tiết các kỹ thuật phân tích

- Persistency mechanism
  - Granularity
  - Volume
  - Duration
  - Access mechanism
  - Access frequency (creation/deletion, update, read)
  - Reliability
- Inter-process Communication mechanism
  - Latency
  - Synchronicity
  - Message size
  - Protocol

Đặc tả chi tiết các kỹ thuật phân tích

- Legacy interface mechanism
  - Latency
  - Duration
  - Access mechanism
  - Access frequency
- Security mechanism
  - Data granularity
  - User granularity
  - Security rules
  - Privilege types
- Others