



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY



Viện Toán Ứng dụng và Tin học

Phân tích thiết kế hệ thống

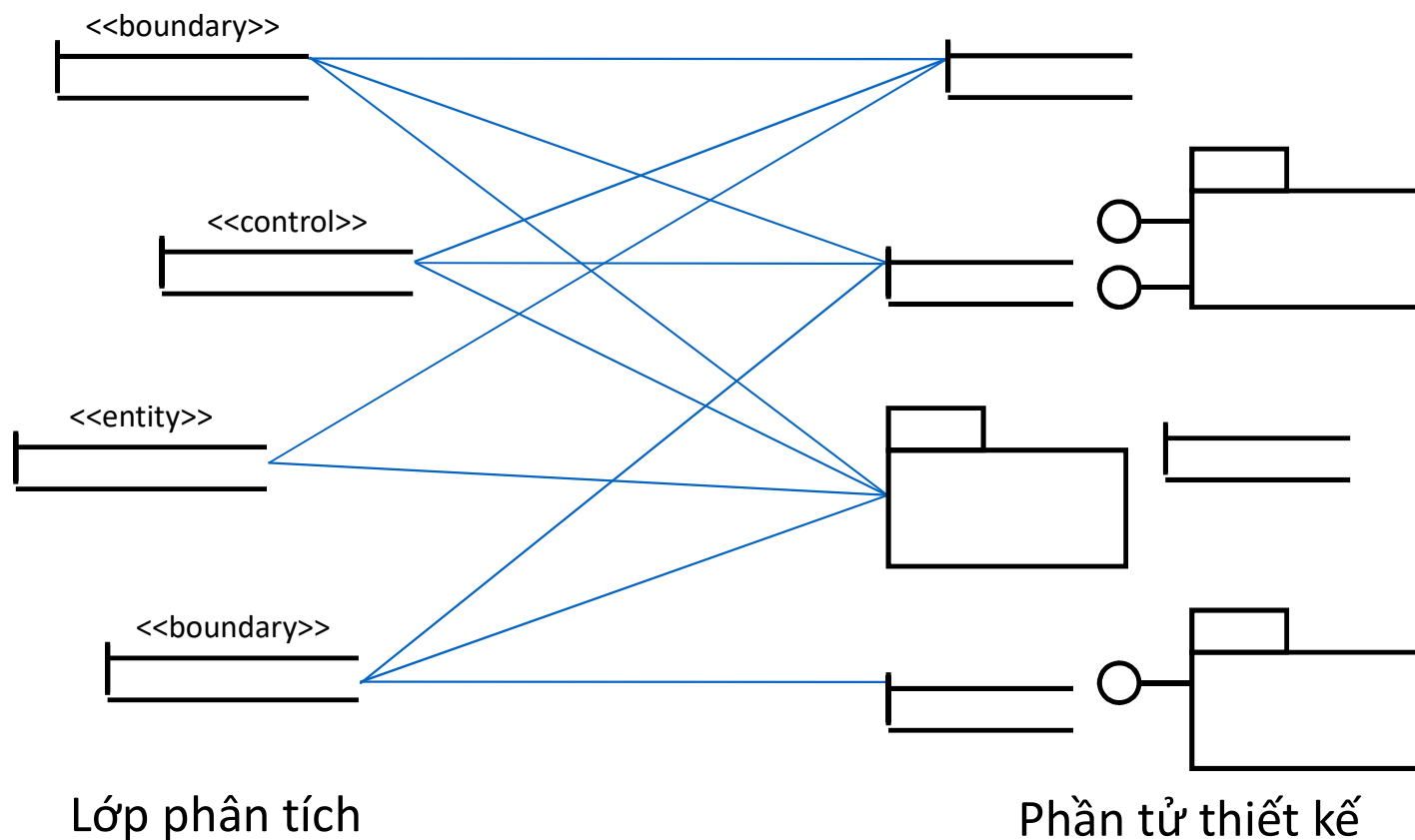
Nhận diện các phần tử thiết kế



Nội dung

- Nhận diện các lớp và hệ thống con
- Nhận diện các giao diện hệ thống con
- Nhận diện cơ hội tái sử dụng
- Cập nhật tổ chức của mô hình thiết kế
- Mẫu thiết kế

Chuyển đổi từ các lớp phân tích sang phần tử thiết kế



Nhận diện các lớp thiết kế

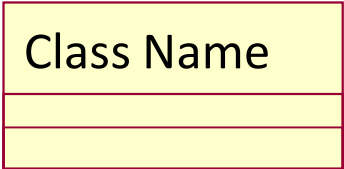
- Lớp phân tích được ánh xạ trực tiếp thành phần tử thiết kế nếu:
 - Nó là lớp đơn giản
 - Nó thể hiện một khái niệm logic đơn
- Lớp phân tích phức tạp có thể
 - Cắt vào nhiều lớp nhỏ hơn
 - Trở thành gói/package
 - Trở thành hệ thống con/subsystem
 - Hay tổ hợp của các thành phần khác.



Lớp và gói

■ Lớp?

- Mô tả tập các đối tượng cùng trách nhiệm, quan hệ, hoạt động, thuộc tính và ngữ nghĩa

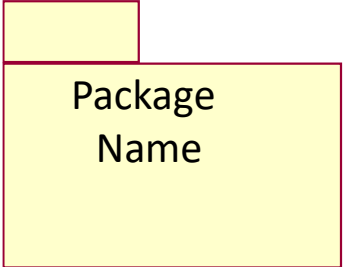


Class Name

A diagram of a class box, which is a rectangle divided into three horizontal compartments. The top compartment is labeled 'Class Name'. The middle and bottom compartments are currently empty.

■ Gói?

- Cơ chế mục đích chung để tổ chức các phần tử vào nhóm
- Phần tử mô hình mà có thể chứa các phần tử mô hình khác

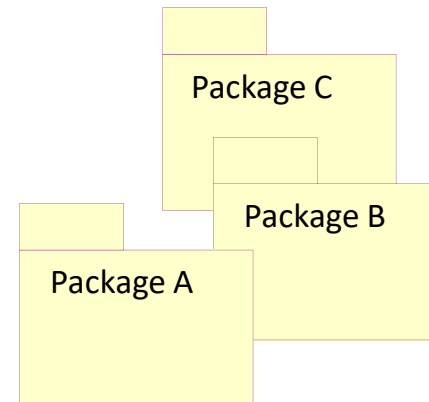


Package Name

A diagram of a package box, which is a rectangle divided into two horizontal compartments. The top compartment is a small empty box. The bottom compartment is labeled 'Package Name'.

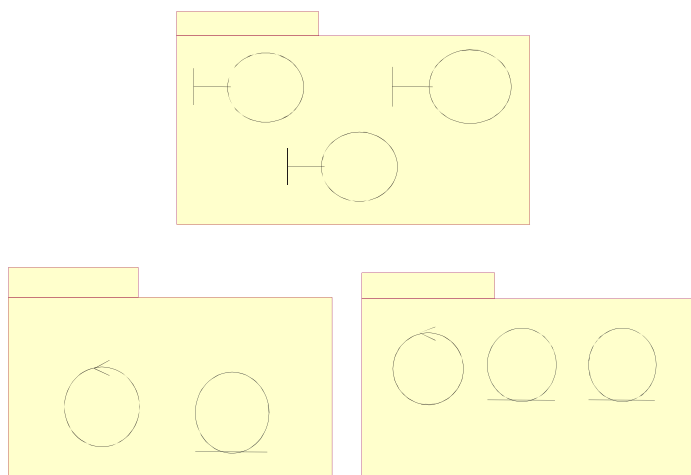
Nhóm các lớp thiết kế vào gói

- Chúng ta tổ chức các phần tử vào các gói dựa trên nhiều tiêu chí như:
 - Các đơn vị cấu hình, cài đặt
 - Phân bổ tài nguyên trong các nhóm phát triển
 - Phản ánh các dạng người dùng
 - Thể hiện các sản phẩm hay dịch vụ đã tồn tại mà hệ thống sử dụng



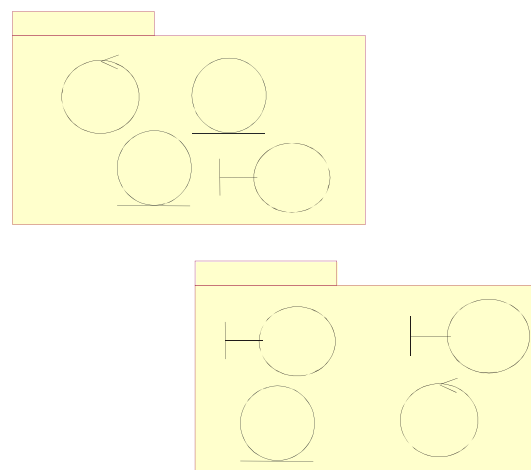
Mẹo tạo gói – với các lớp biên

Nếu các giao diện thường xuyên bị thay đổi



Các lớp biên được đặt cùng vào một gói

Nếu các giao diện không thường xuyên bị thay đổi



Lớp biên được gói cùng các lớp chức năng liên quan

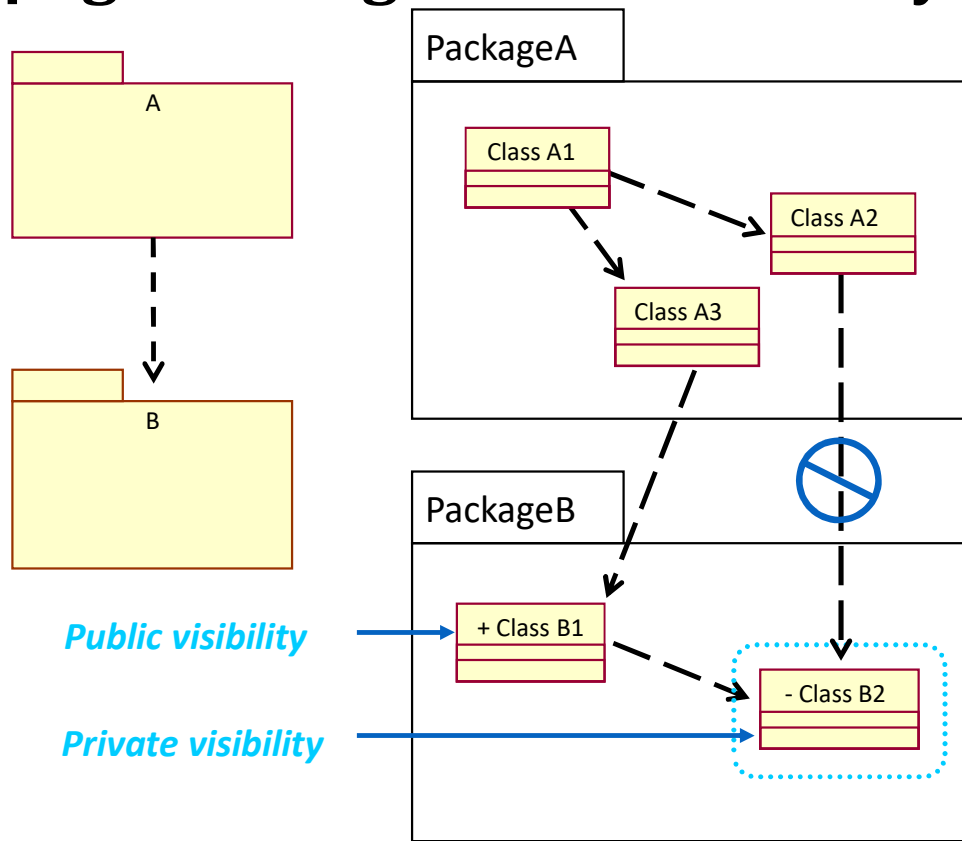
Hướng dẫn tạo gói/package – các lớp liên quan về chức năng

- Tiêu chuẩn để xác định nếu các gói liên quan về chức năng:
 - Các thay đổi về hành vi hay cấu trúc của một lớp dẫn tới thay đổi ở lớp khác
 - Loại bỏ 1 lớp dẫn tới ảnh hưởng lớp khác
 - Hai đối tượng tương tác với một số lớp thông báo hoặc có trao đổi phức tạp
 - Lớp biên có thể liên quan chức năng với 1 lớp thực thể cụ thể nếu chức năng của lớp biên là thể hiện lớp thực thể
 - Hai lớp tương tác với nhau hoặc bị ảnh hưởng bởi các thay đổi của cùng một tác nhân
 - Hai lớp có quan hệ với nhau
 - Một lớp tạo thể hiện của lớp khác

HD tạo gói/package – các lớp liên quan về chức năng

- Tiêu chuẩn để xác định hai lớp nên được đặt ở những gói riêng biệt:
 - Hai lớp liên quan tới 2 tác nhân khác nhau không nên đặt chung vào cùng một gói
 - Lớp lựa chọn và lớp bắt buộc không nên đặt chung vào cùng một gói

Phụ thuộc giữa các gói: tính nhìn thấy - visibility

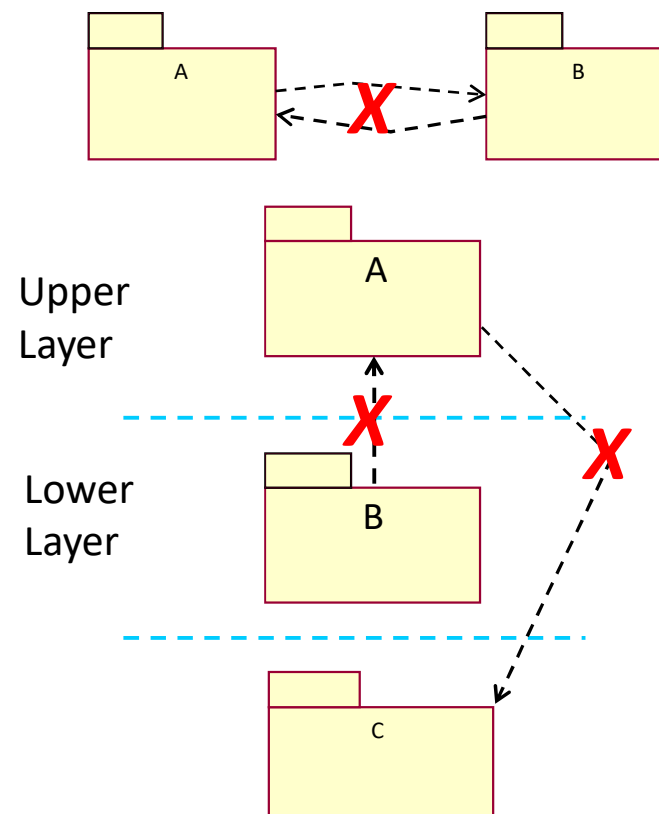


Chỉ các lớp public có thể được tham chiếu tới từ ngoài gói của nó

Nguyên tắc đóng gói

Phụ thuộc gói – tham chiếu

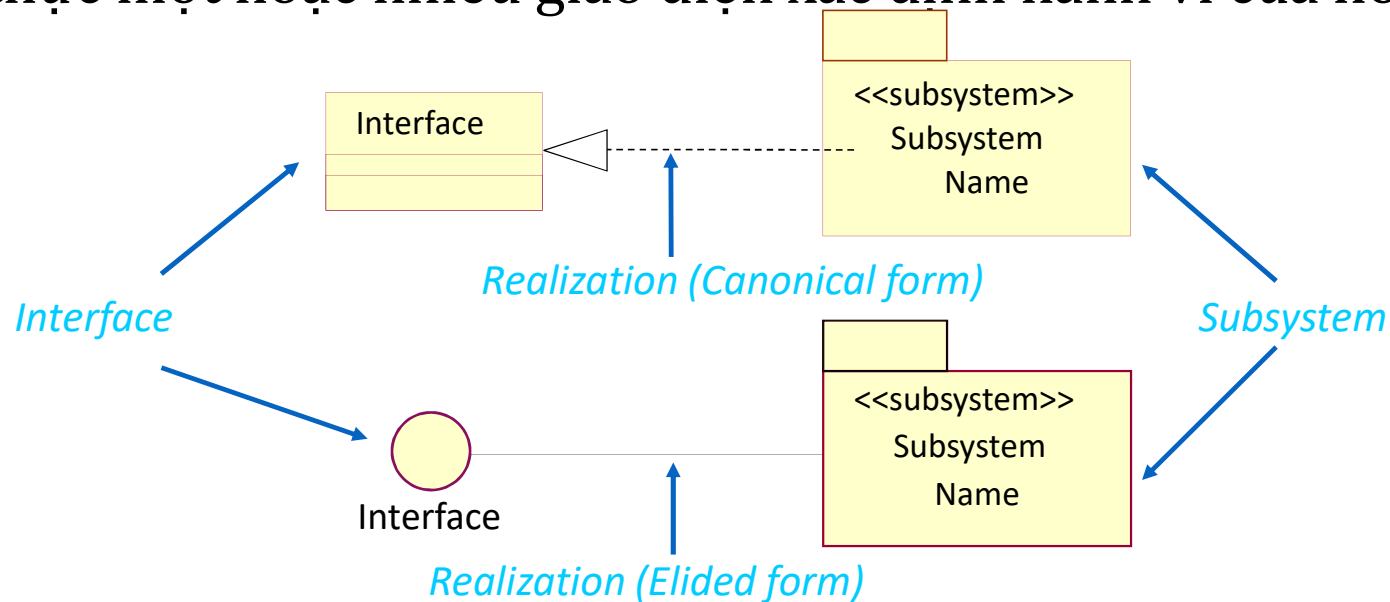
- Các gói không nên tham chiếu lẫn nhau
- Các gói ở tầng thấp không nên phụ thuộc vào tầng cao
- Nói chung các phụ thuộc không nên vượt tầng



X = Coupling violation

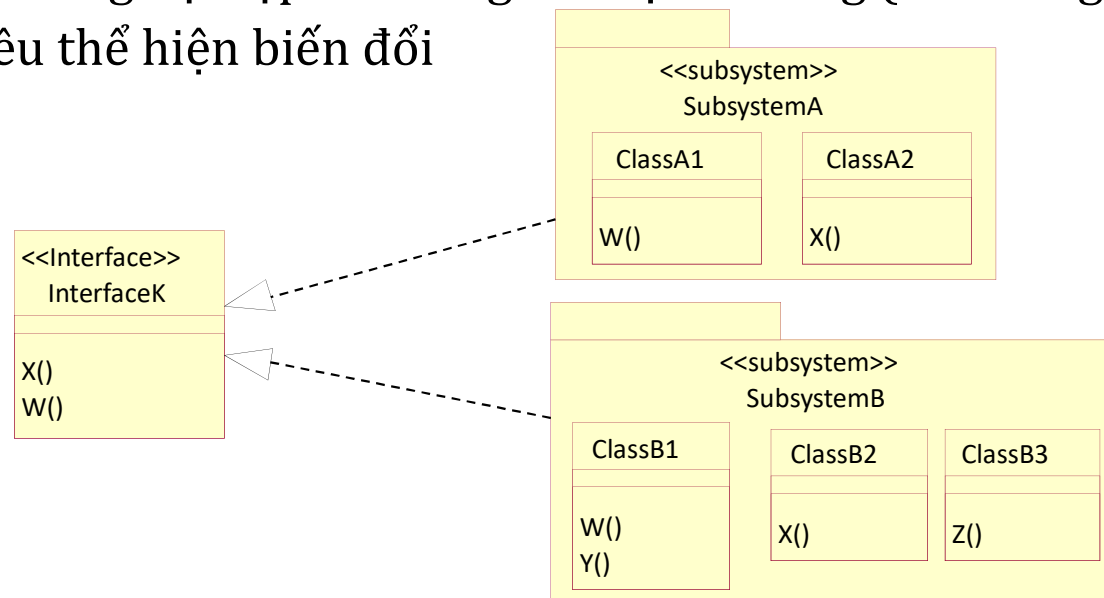
Hệ thống con và giao diện

- Tương tự gói (có thể chứa các phần tử mô hình khác) và tương tự lớp (có hành vi)
- Hiện thực một hoặc nhiều giao diện xác định hành vi của nó



Hệ thống con và giao diện

- Hệ thống con:
 - Đóng gói các hành vi
 - Thể hiện khả năng độc lập với các giao diện rõ ràng (khả năng sử dụng lại)
 - Mô hình nhiều thể hiện biến đổi



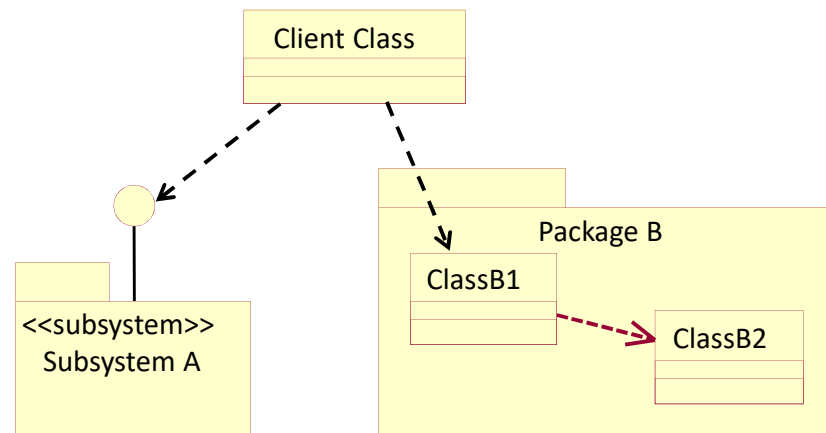
Gói hay hệ thống con

Hệ thống con

- Cung cấp hành vi
- Đóng gói hoàn chỉnh hành vi của nó
- Dễ dàng thay thế

Gói

- Không cung cấp hành vi
- Không đóng gói hoàn chỉnh hành vi của nó
- Không dễ thay thế



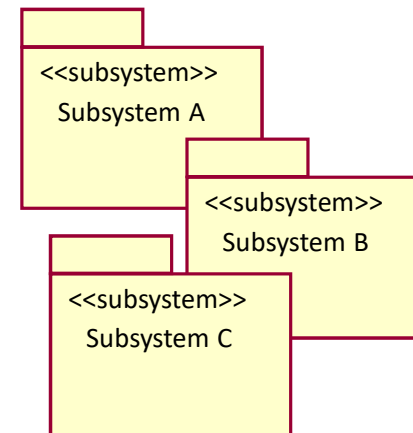
Đóng gói là đặc điểm chính

Sử dụng hệ thống con

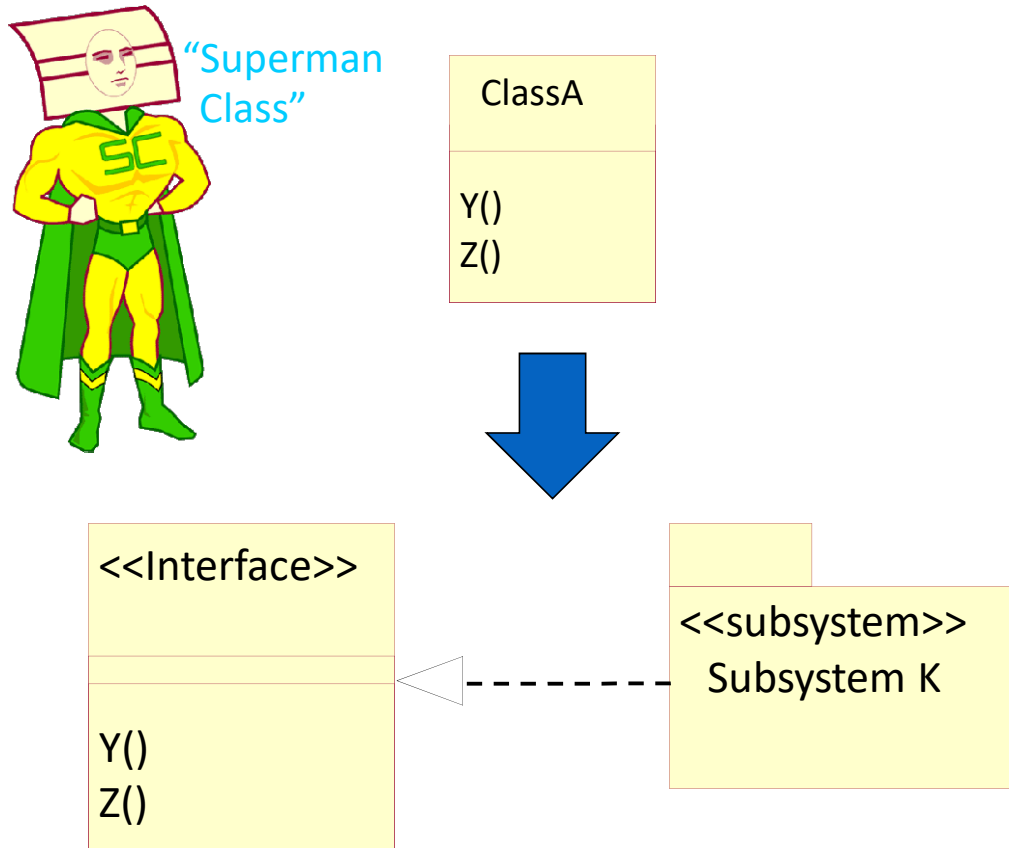
- Các hệ thống con có thể được sử dụng để phân hoạch hệ thống vào các phần có thể độc lập:
 - Đặt hàng, cấu hình hay phân phối
 - Phát triển, khi các giao diện không thay đổi
 - Triển khai trên tập các điểm tính toán
 - Thay đổi mà không làm ảnh hưởng các phần khác
- Các hệ thống con cũng có thể được sử dụng để:
 - Phân hoạch hệ thống vào các đơn vị có thể cung cấp tính bảo mật chặt với các tài nguyên chính
 - Thể hiện các sản phẩm đã có hay các hệ thống ngoài trong quá trình thiết kế (thí dụ các thành phần)

Nhận diện các hệ thống con

- Lớp phân tích có thể chuyển vào hệ thống con:
 - Lớp cung cấp các dịch vụ hay tiện ích phức tạp
 - Lớp biên (giao diện người dùng và giao diện với hệ thống ngoài)
- Các sản phẩm hiện có hay các hệ thống ngoài trong quá trình thiết kế (như các thành phần):
 - Phần mềm trao đổi thông tin
 - Hỗ trợ truy cập CSDL
 - Các dạng và cấu trúc dữ liệu
 - Các tiện ích chung
 - Các sản phẩm ứng dụng cụ thể



Nhận diện hệ thống con



Nhận diện giao diện hệ thống con

- Mục đích

- Để nhận diện các giao diện của hệ thống con dựa trên các trách nhiệm của chúng

- Các bước

- Nhận diện tập các giao diện ứng cử cho tất cả các hệ thống con.
- Tìm kiếm sự tương tự giữa các giao diện.
- Xác định các phụ thuộc giao diện.
- Ánh xạ các giao diện tới các hệ thống con.
- Xác định hành vi của các giao diện.
- Đóng gói các giao diện.

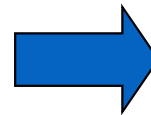
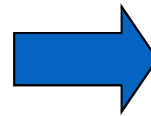
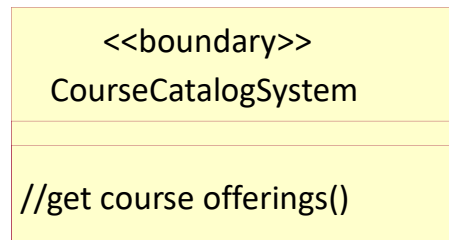
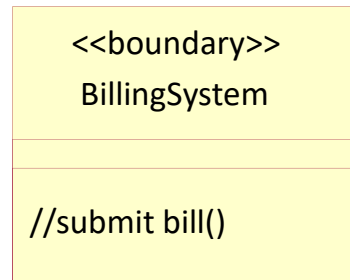
Chỉ dẫn giao diện

- Tên giao diện
 - Phản ánh vai trò của nó
- Mô tả giao diện
 - Truyền tải các trách nhiệm
- Xác định các chức năng
 - Tên nên phản ánh các kết quả
 - Mô tả chức năng làm gì, tất cả tham số và kết quả
- Tài liệu giao diện
 - Gói cung cấp thông tin: các biểu đồ tuần tự và trạng thái, kế hoạch test, vv.

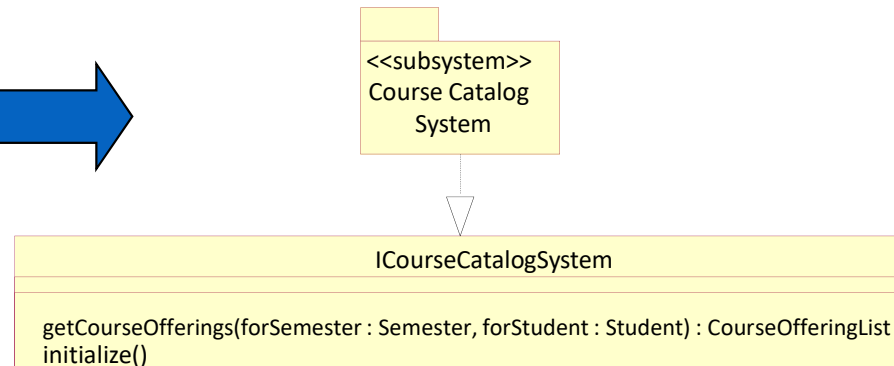
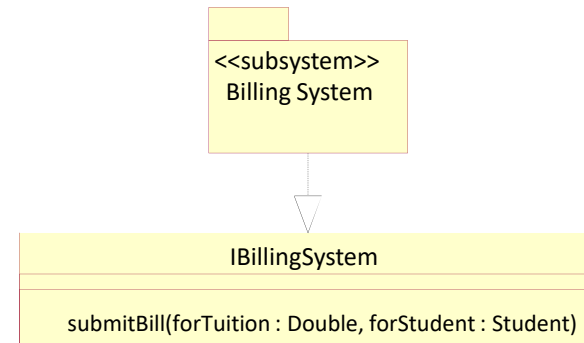


Thí dụ

Analysis

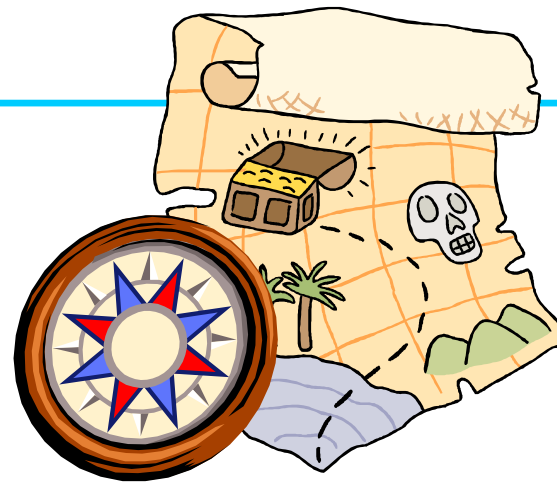


Design

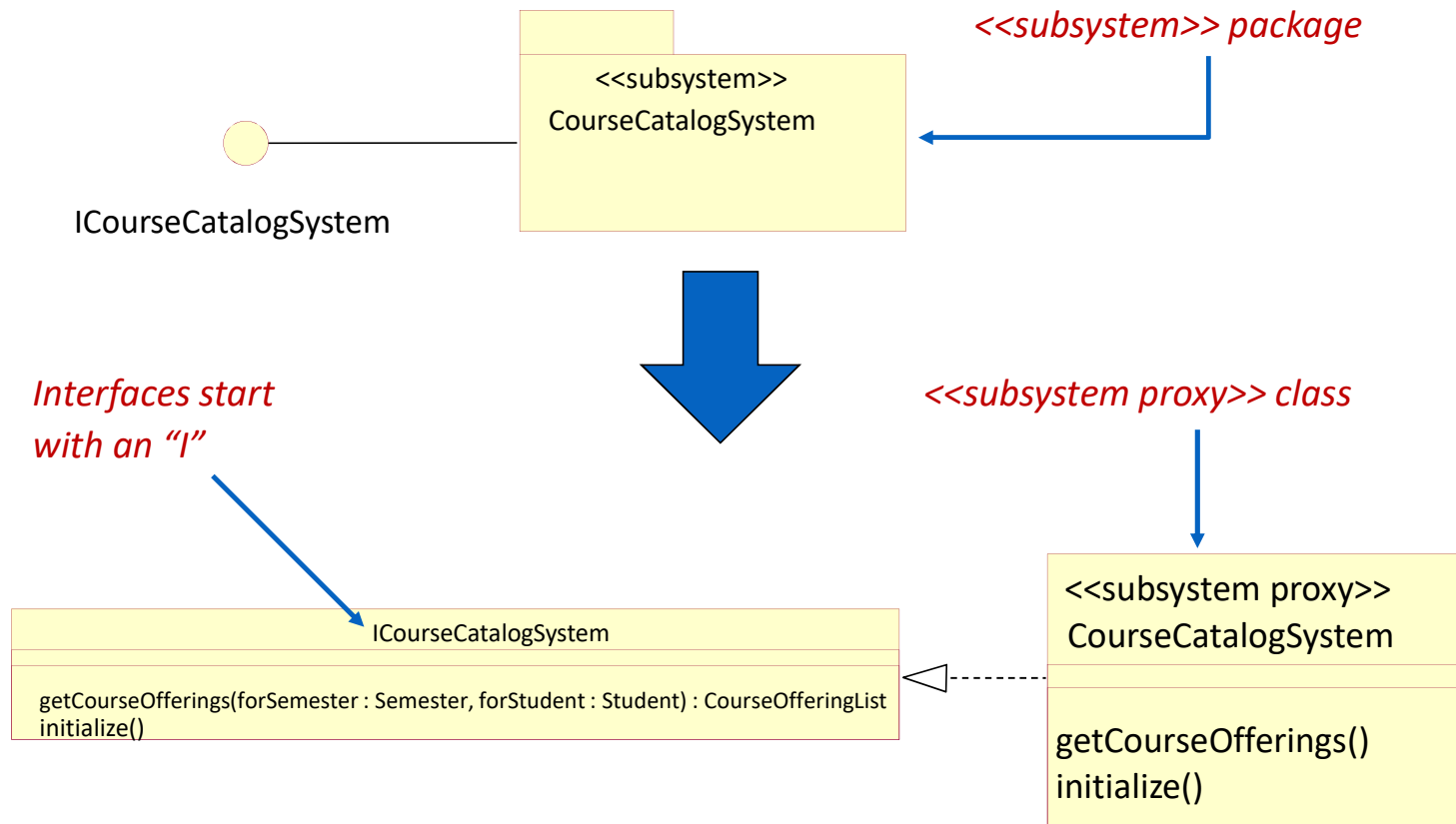


Thí dụ

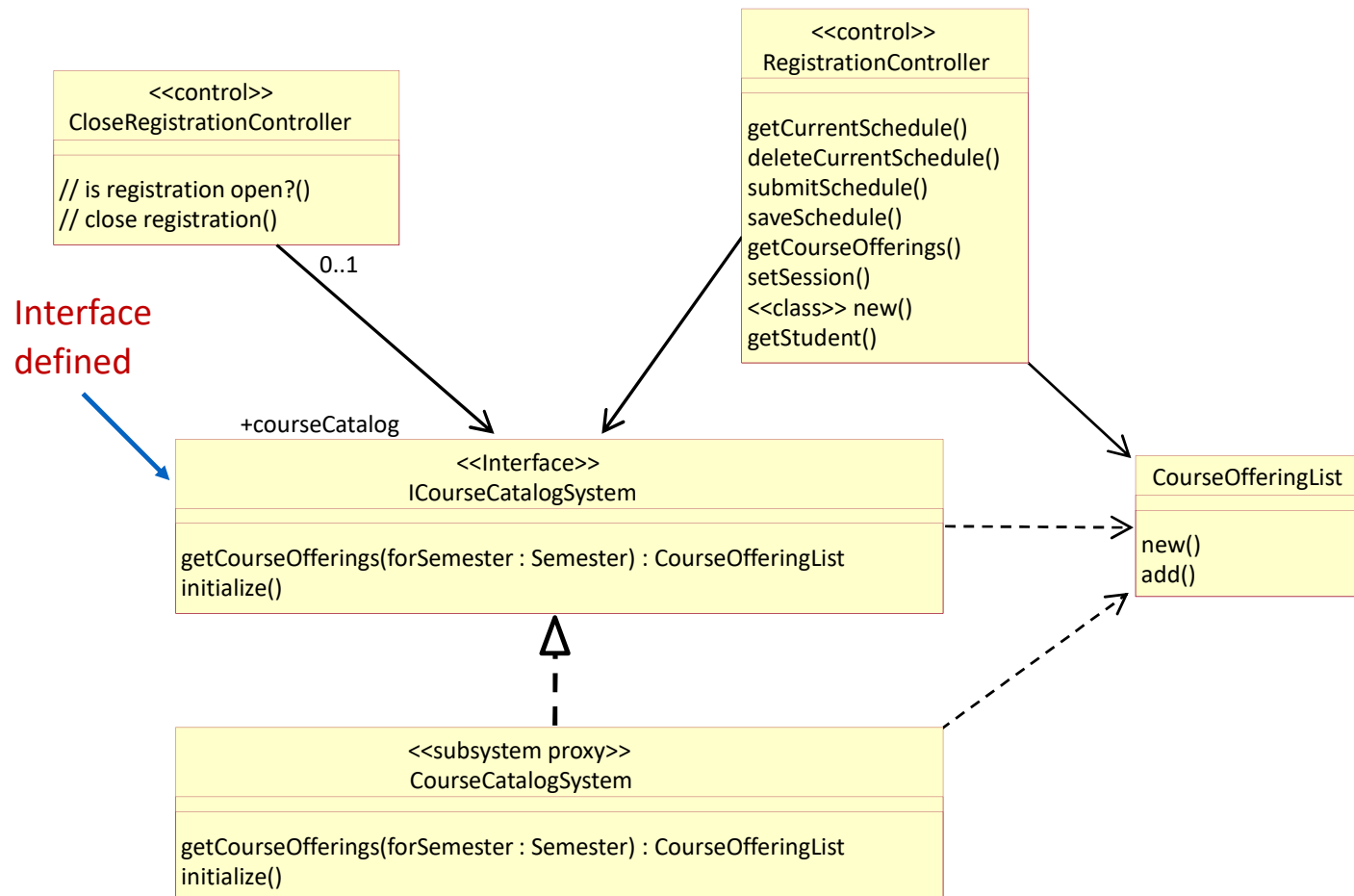
Analysis Class	Design Element
CourseCatalogSystem	CourseCatalogSystem Subsystem
BillingSystem	BillingSystem Subsystem
All other analysis classes map directly to design classes	



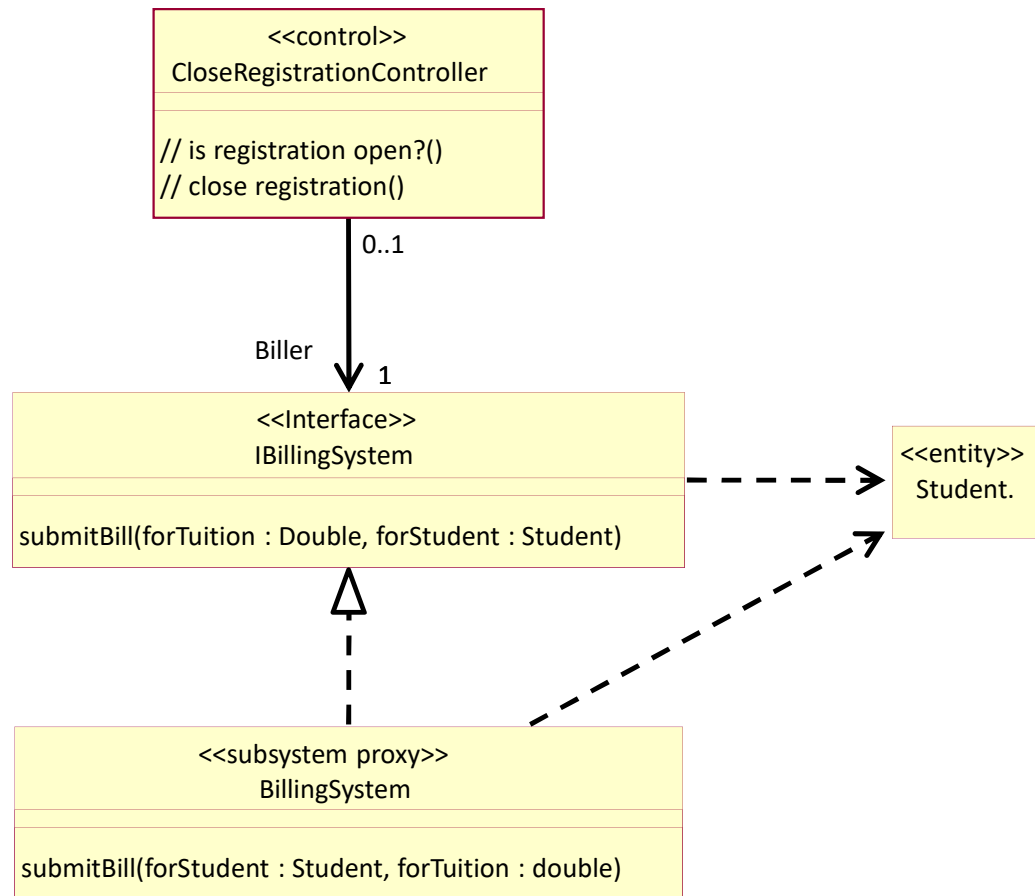
Quy ước mô hình hóa: hệ thống con và giao diện



Thí dụ



Thí dụ



Nhận diện cơ hội sử dụng lại

- Mục đích
 - Để nhận diện liệu các hệ thống con hay các thành phần có thể được sử dụng lại dựa trên các giao diện của chúng.
- Các bước
 - Tìm kiếm các giao diện tương tự
 - Biến đổi các giao diện mới để cải tiến khả năng sử dụng lại
 - Thay thế các giao diện ứng cử với các giao diện đã có
 - Ánh xạ hệ thống con ứng cử vào các thành phần đã có

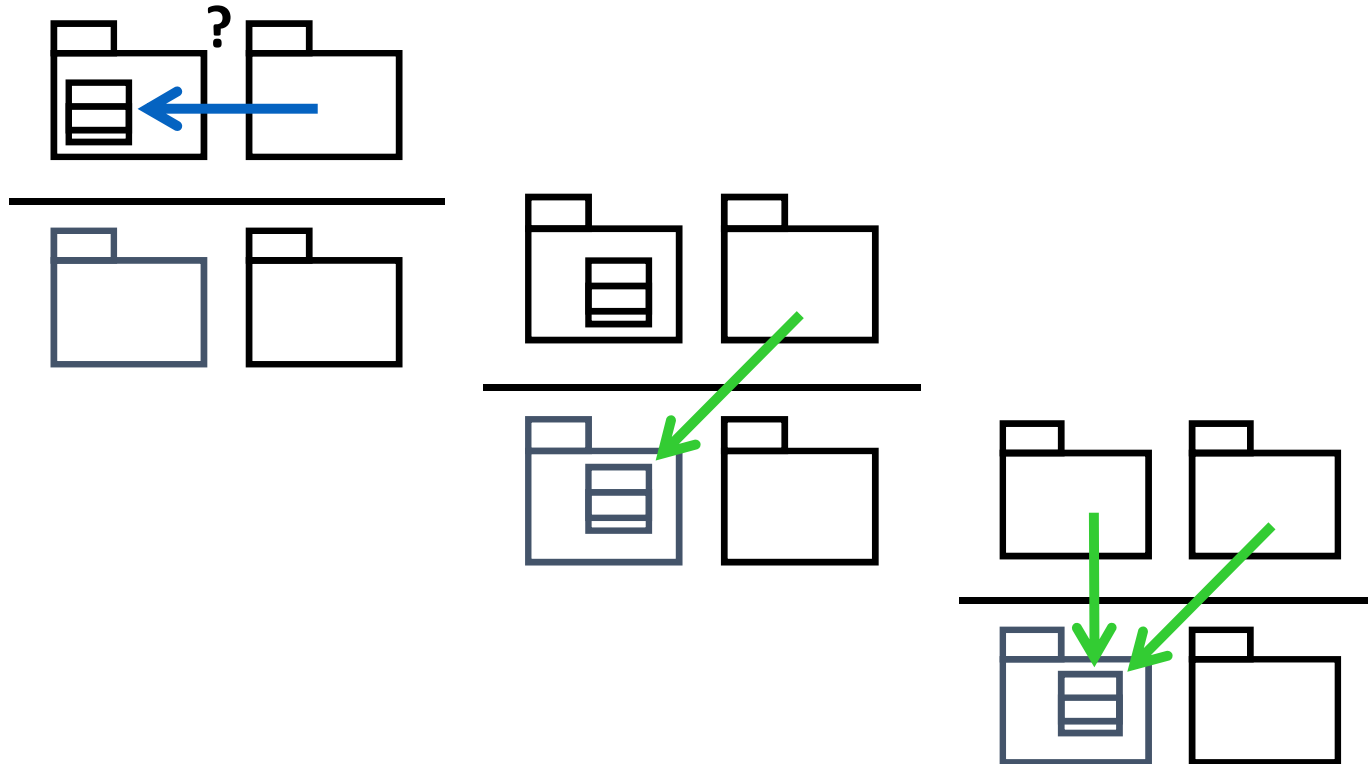


Khả năng sử dụng lại

- Bên trong hệ thống đang phát triển
 - Ghi nhận các thành phần chung giữa các gói hay các hệ thống con
- Bên ngoài hệ thống đang phát triển
 - Các thành phần sẵn dùng thương mại
 - Các thành phần từ ứng dụng được phát triển trước
 - Khôi phục mã các thành phần



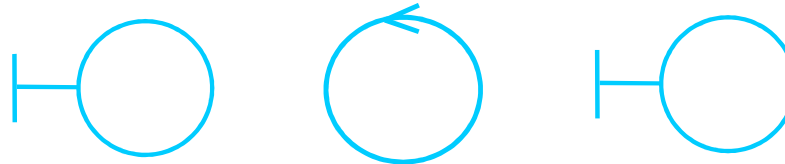
Khả năng sử dụng lại bên trong hệ thống



Cập nhật tổ chức của mô hình thiết kế

- Phần tử thiết kế và kiến trúc

Layer 1



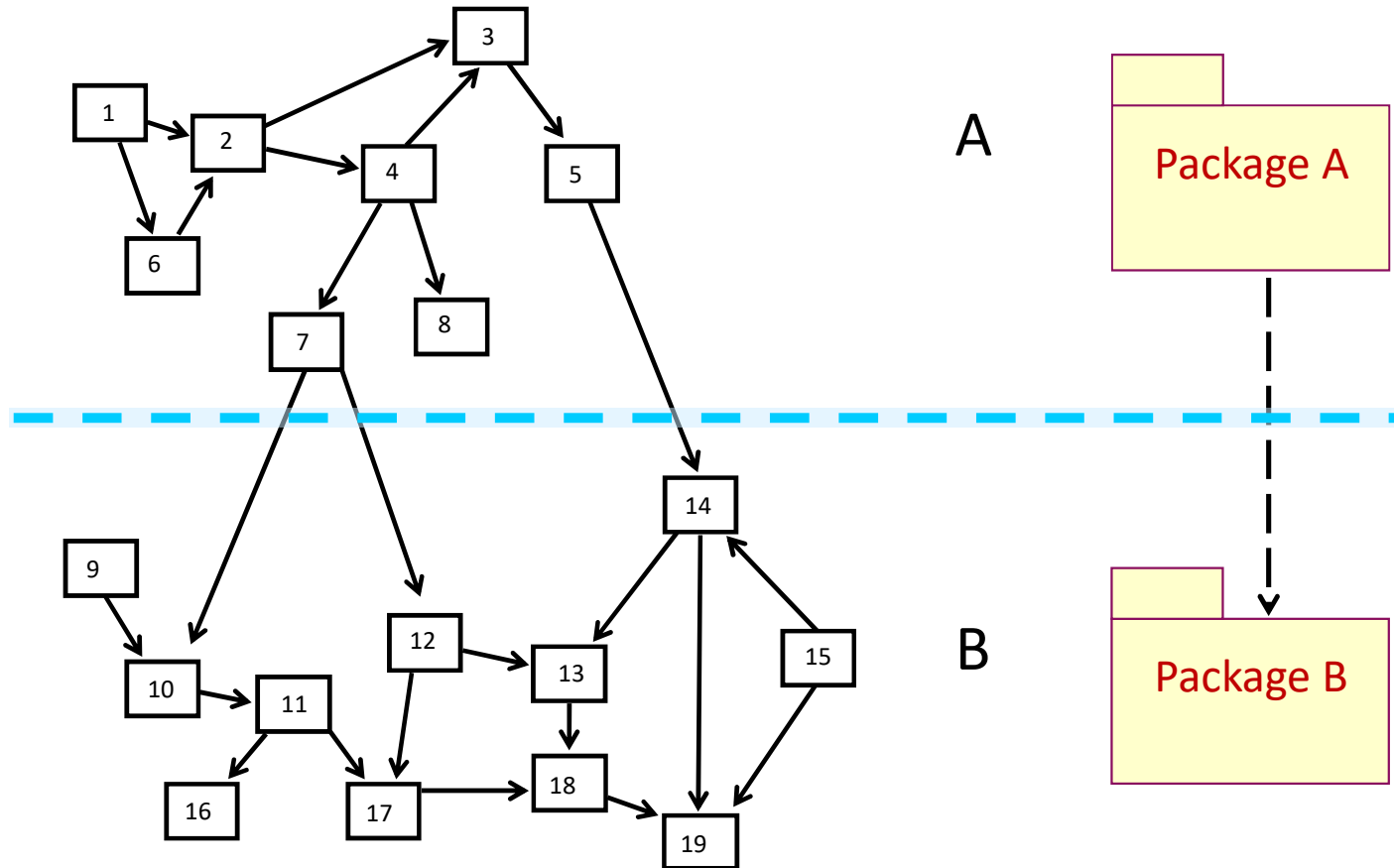
Layer 2



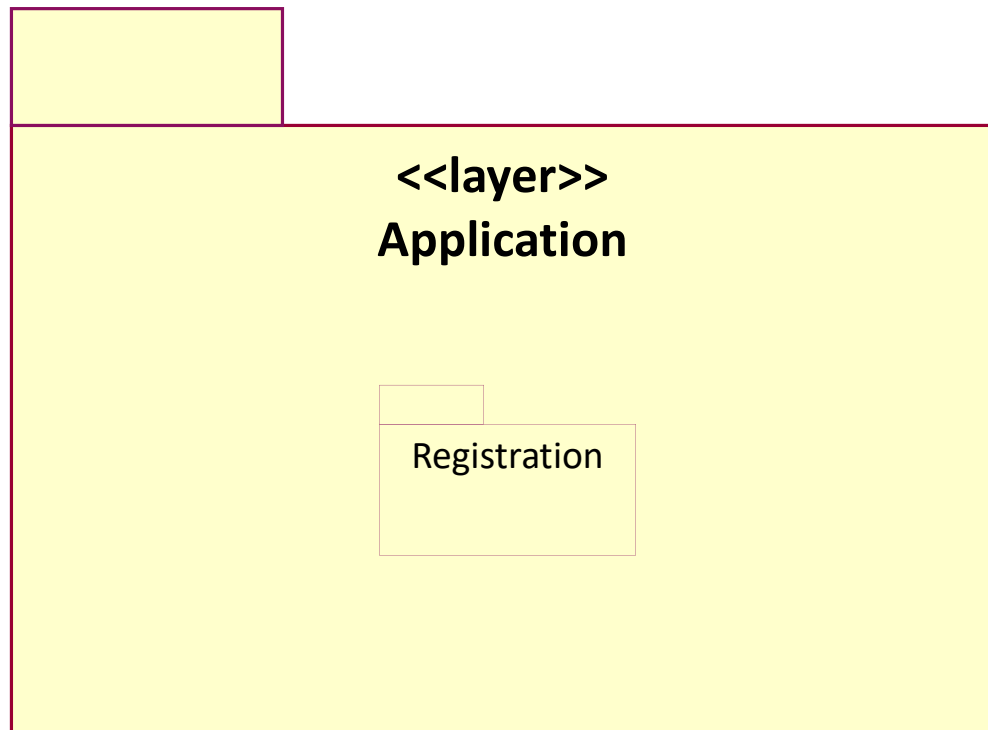
Layer 3



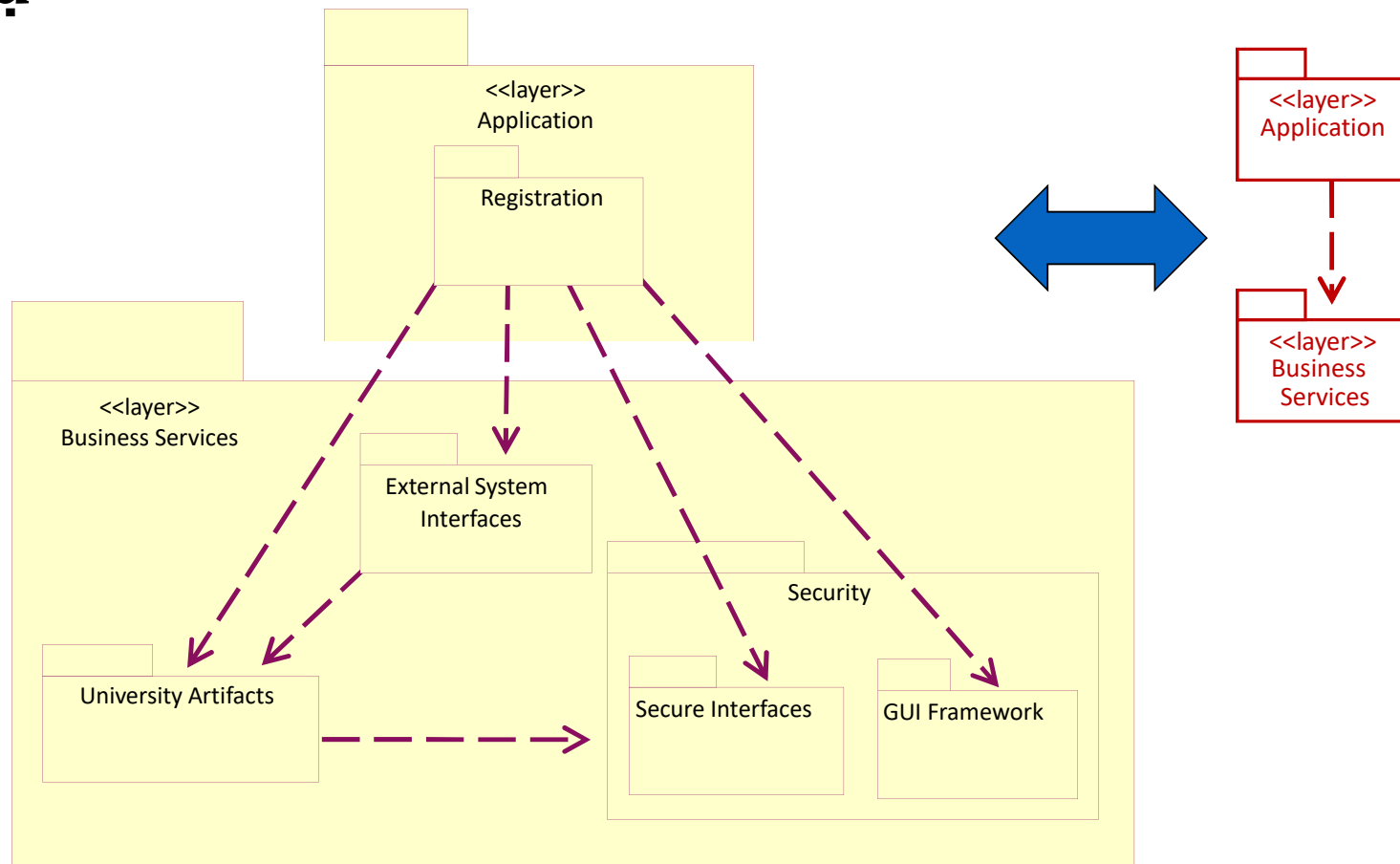
Thí dụ phân hoạch



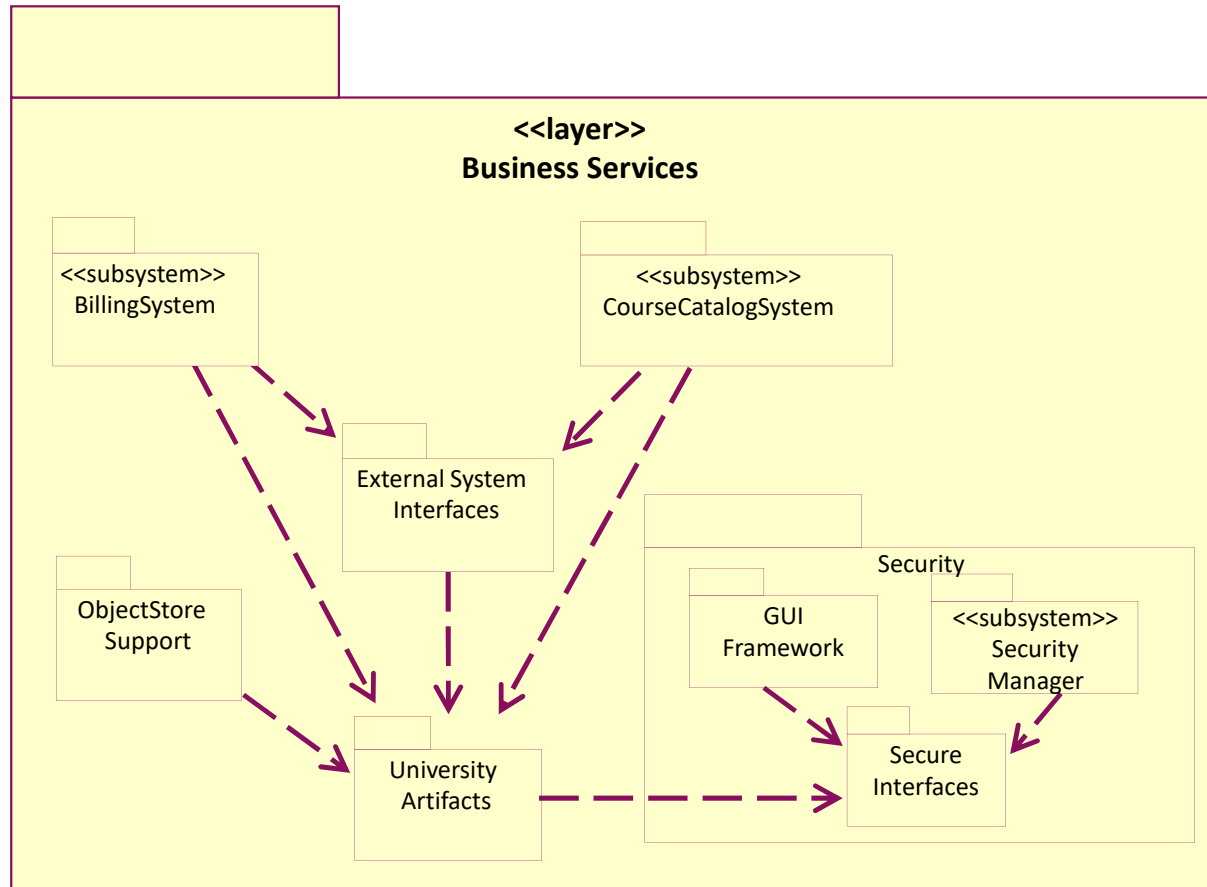
Thí dụ: tầng ứng dụng



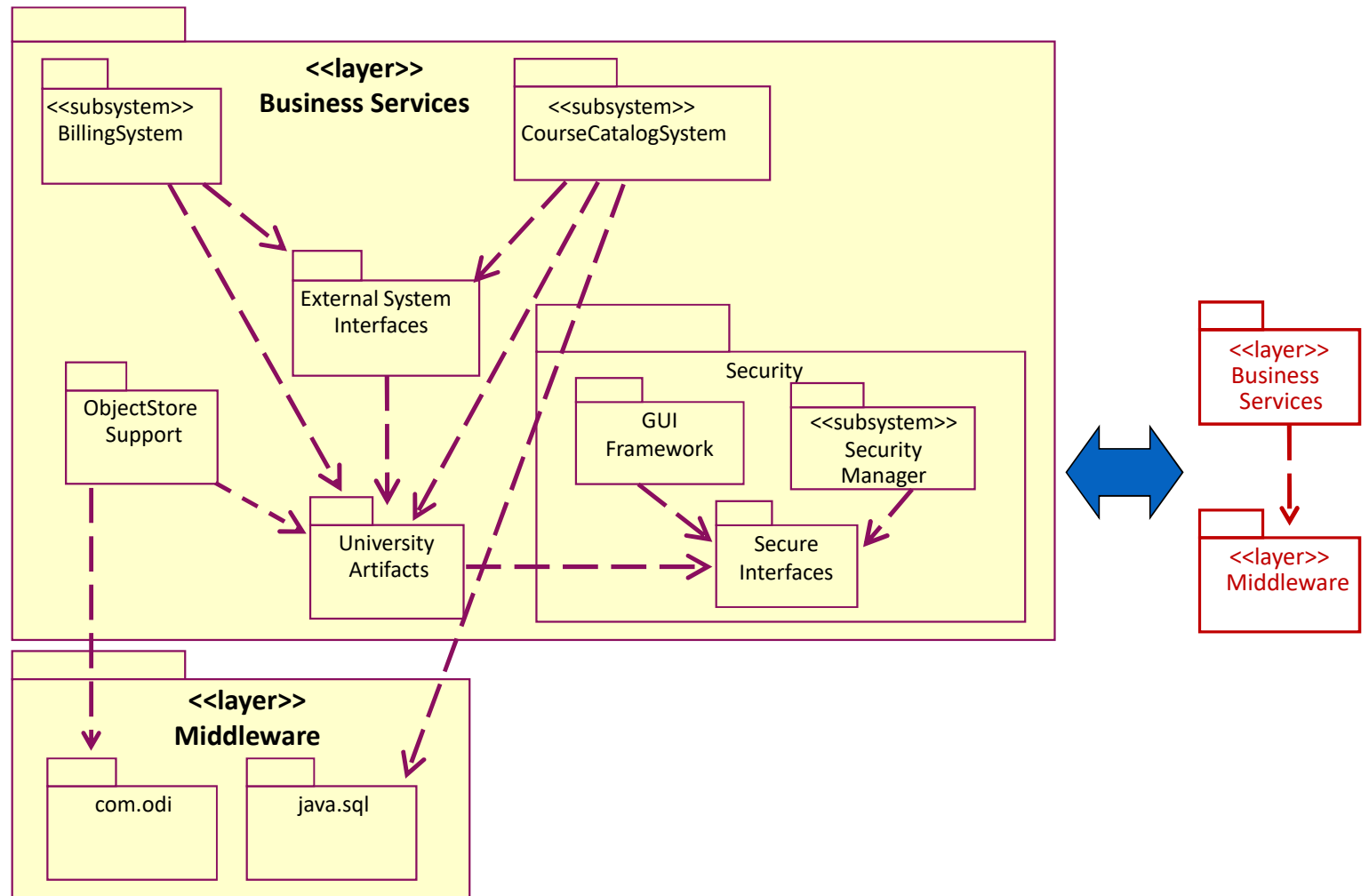
Thí dụ



Thí dụ tầng dịch vụ nghiệp vụ



Thí dụ

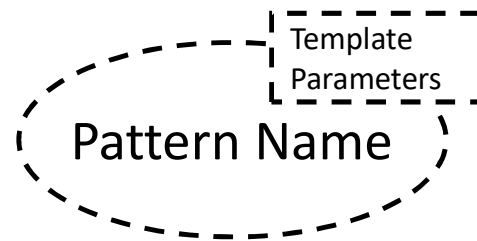


Mẫu thiết kế

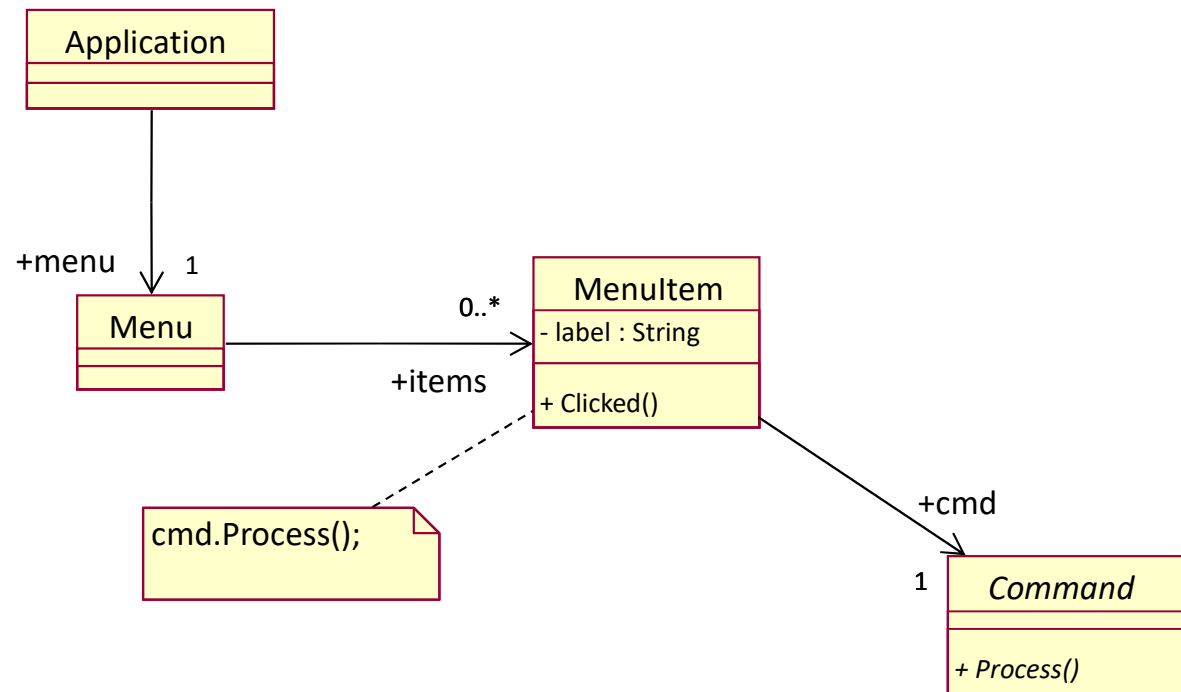
- Khái niệm pattern, framework
- Mẫu: cung cấp 1 giải pháp chung cho một vấn đề chung trong một ngữ cảnh nào đấy
- Mẫu phân tích/thiết kế
 - Cung cấp lời giải cho một vấn đề kỹ thuật phạm vi hẹp
 - Cung cấp một phần của giải pháp
- Khung: xác định cách tiếp cận tổng quát để giải quyết vấn đề. Cung cấp một giải pháp khung mà trong đó chi tiết của nó là các mẫu

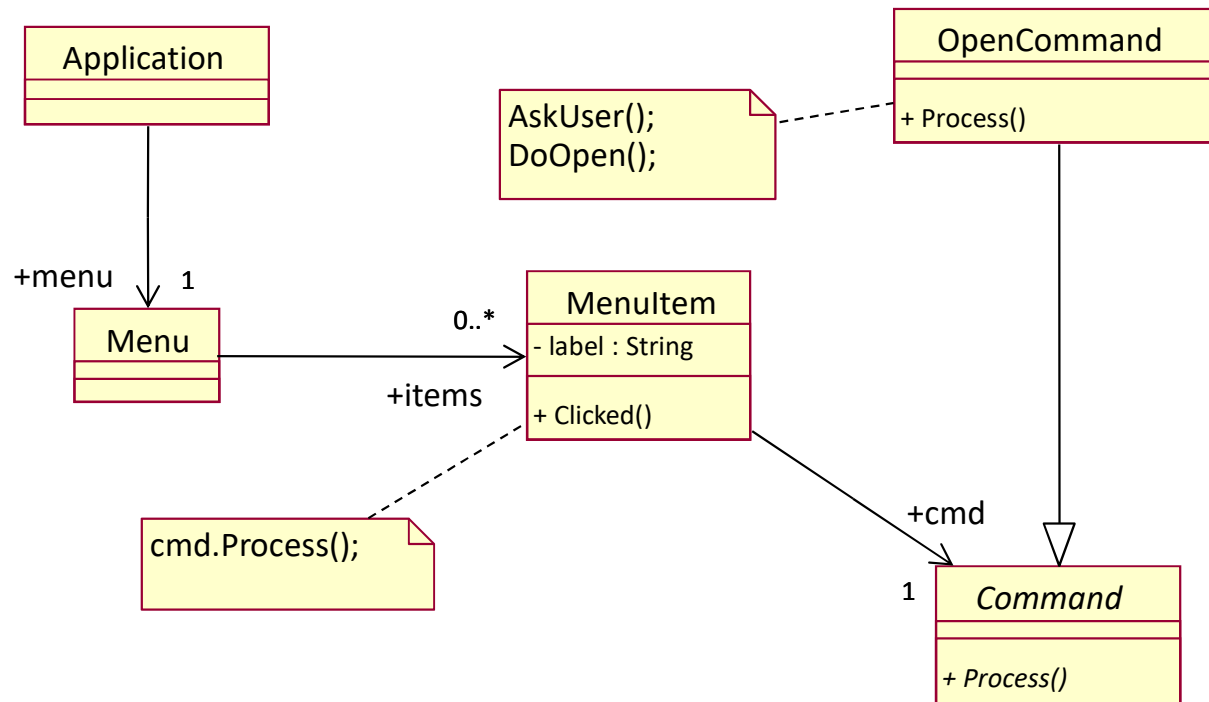
Mẫu thiết kế

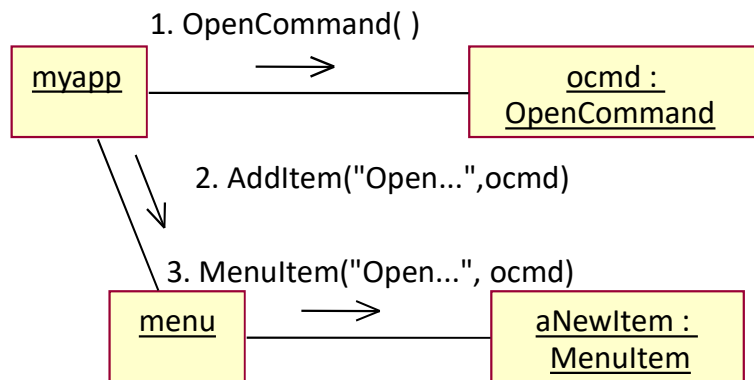
- Mẫu thiết kế cung cấp một lược đồ làm tốt các hệ thống con hay các thành phần phần mềm cũng như quan hệ giữa chúng. Nó mô tả cấu trúc chung của các thành phần mà giải quyết vấn đề thiết kế trong ngữ cảnh cụ thể.



Thí dụ mẫu thiết kế lệnh

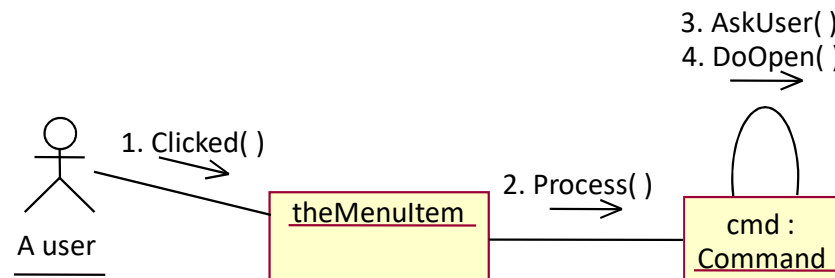


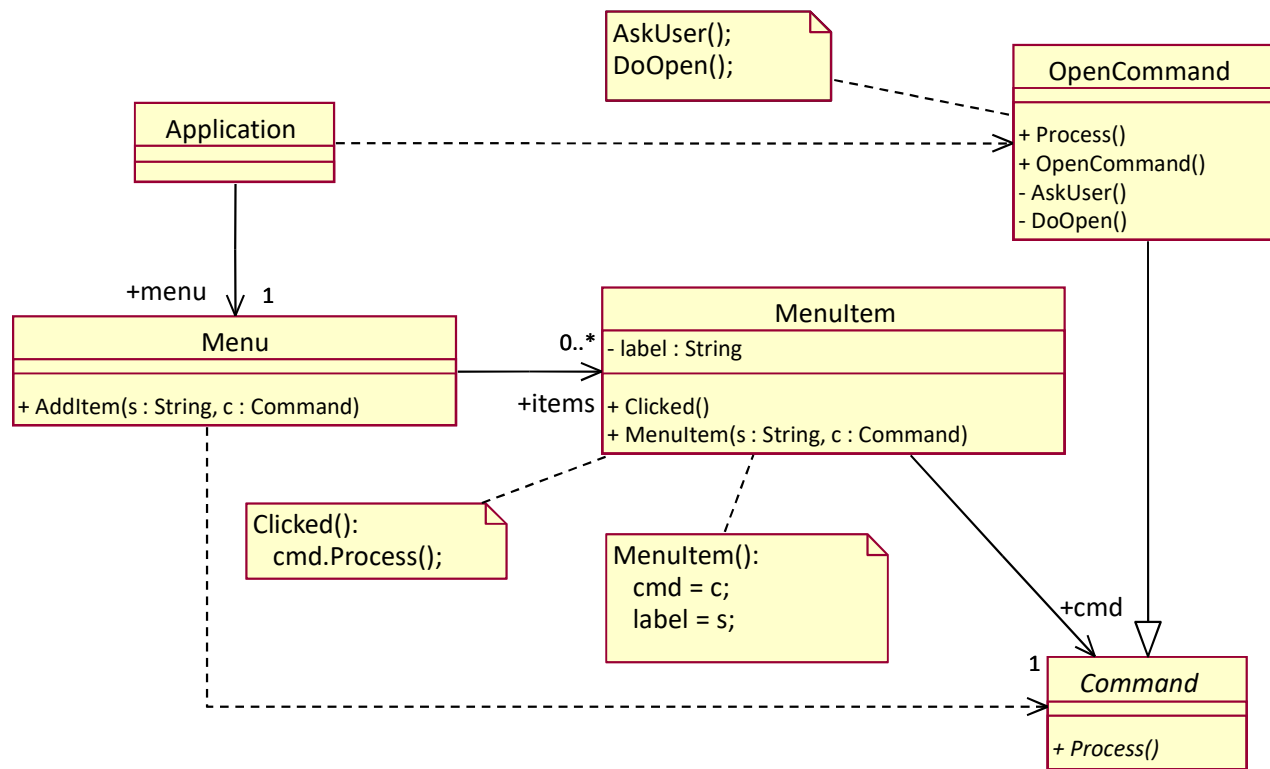


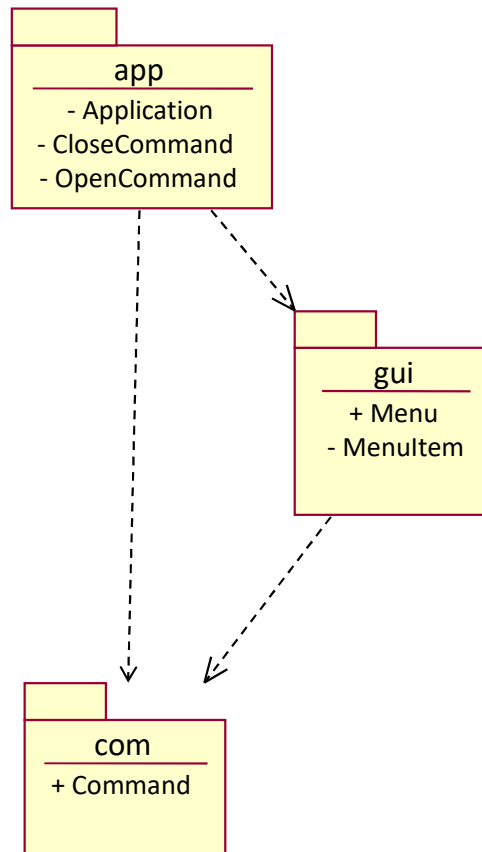


Initialization

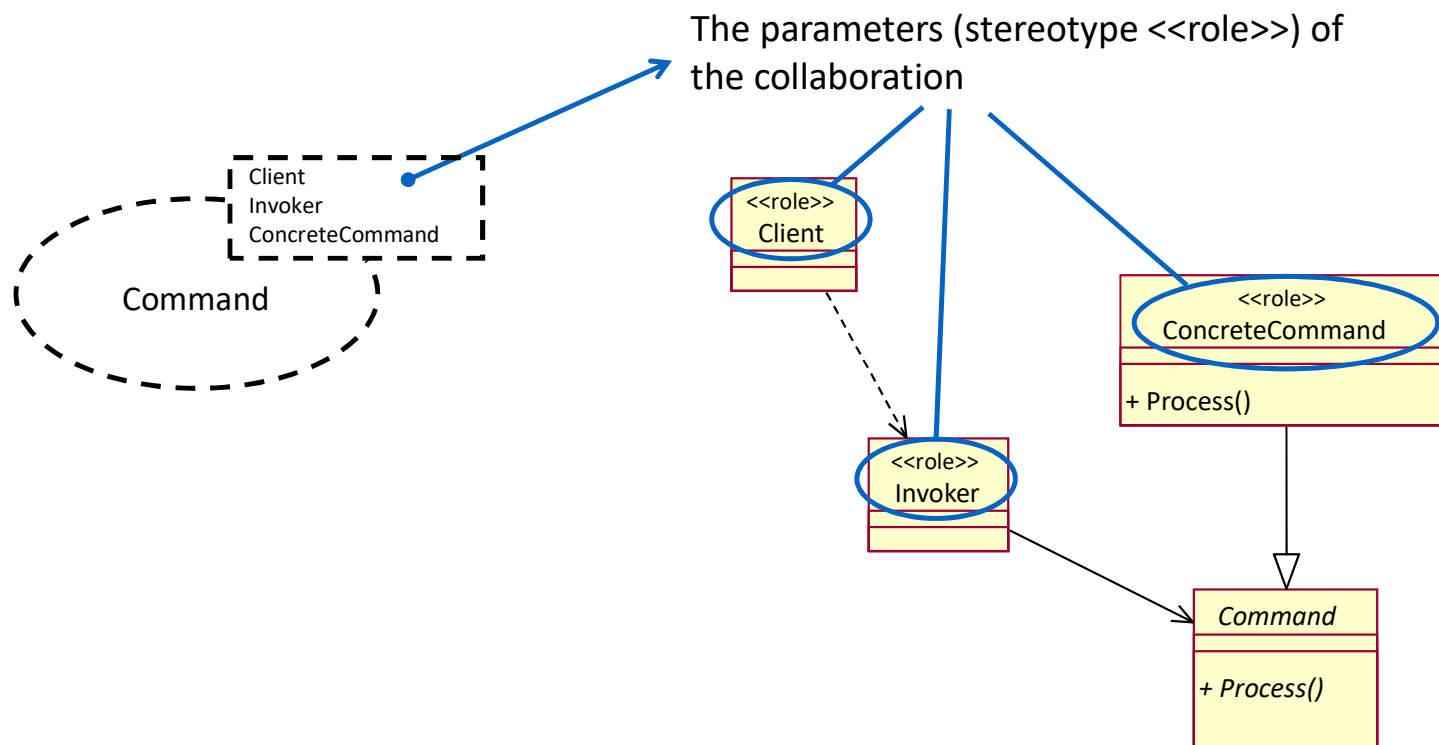
The user selects the *Open...* menu item



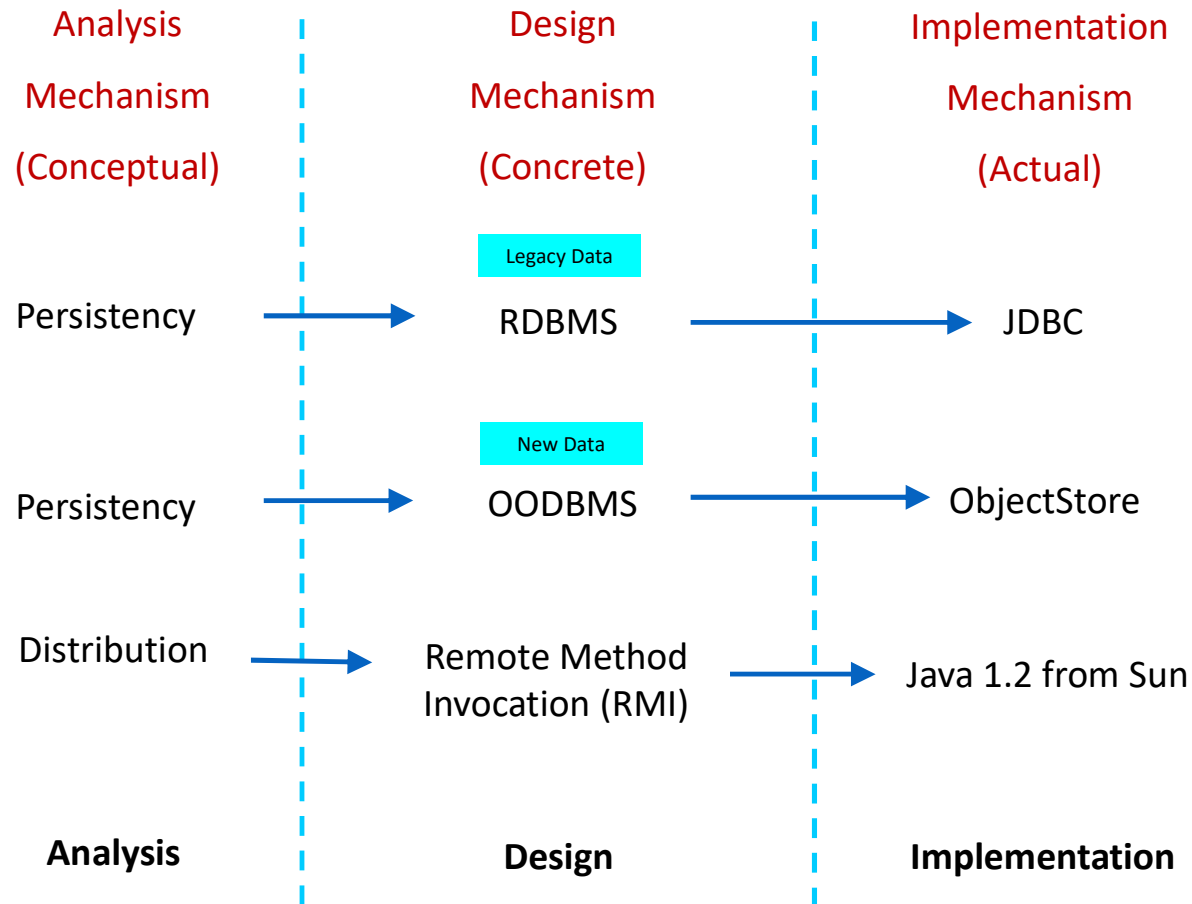




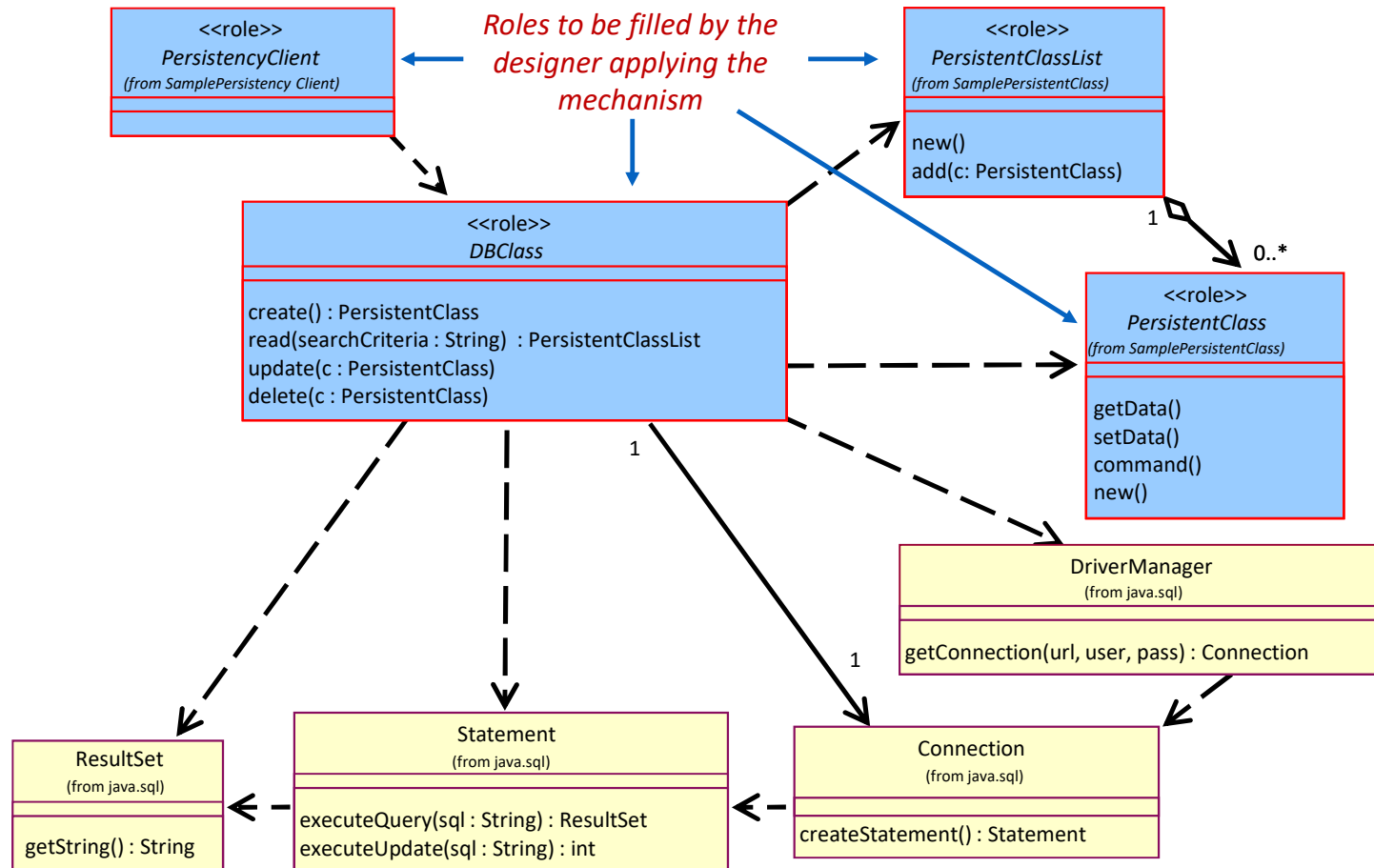
Thể hiện mẫu thiết kế với UML



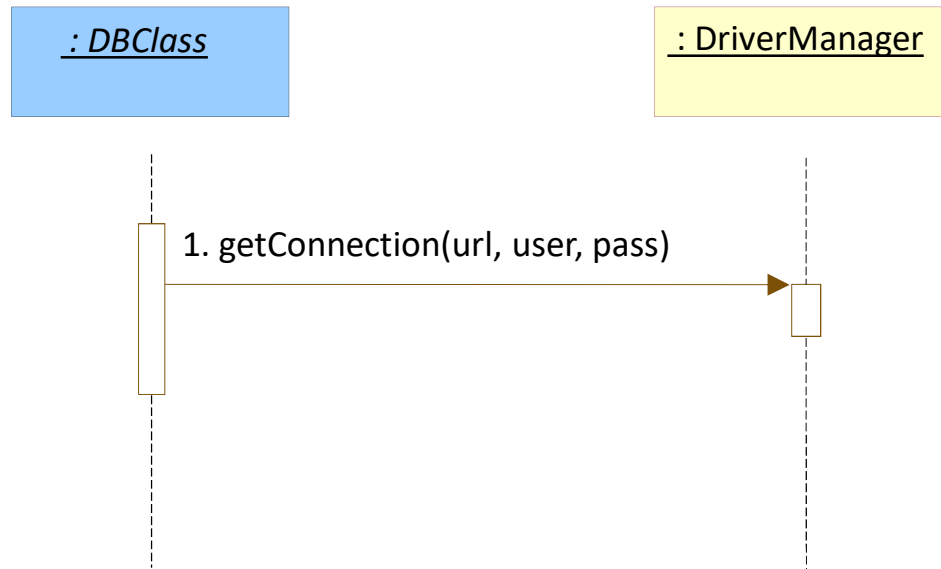
Cơ chế thiết kế và thể hiện



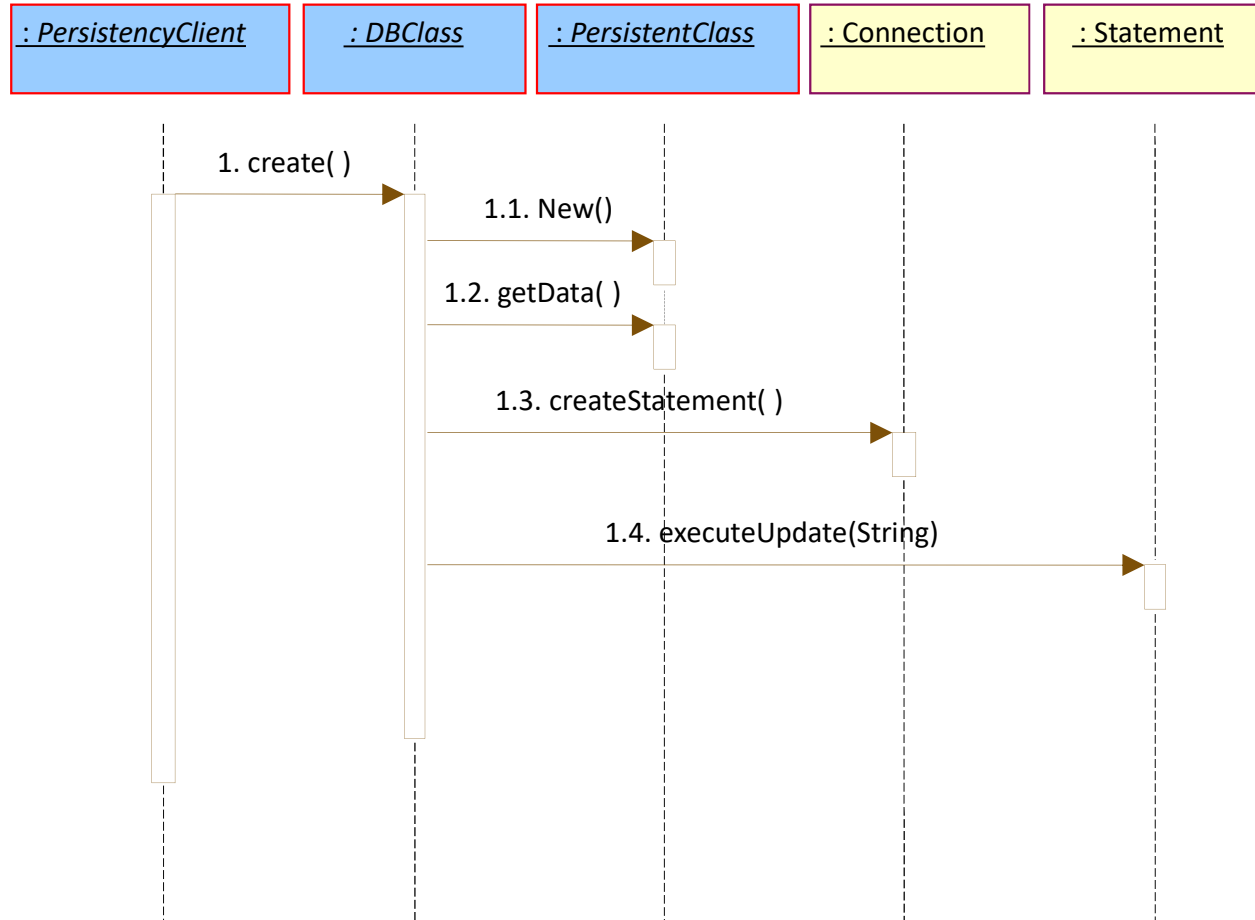
Thí dụ: Persistency: RDBMS: JDBC



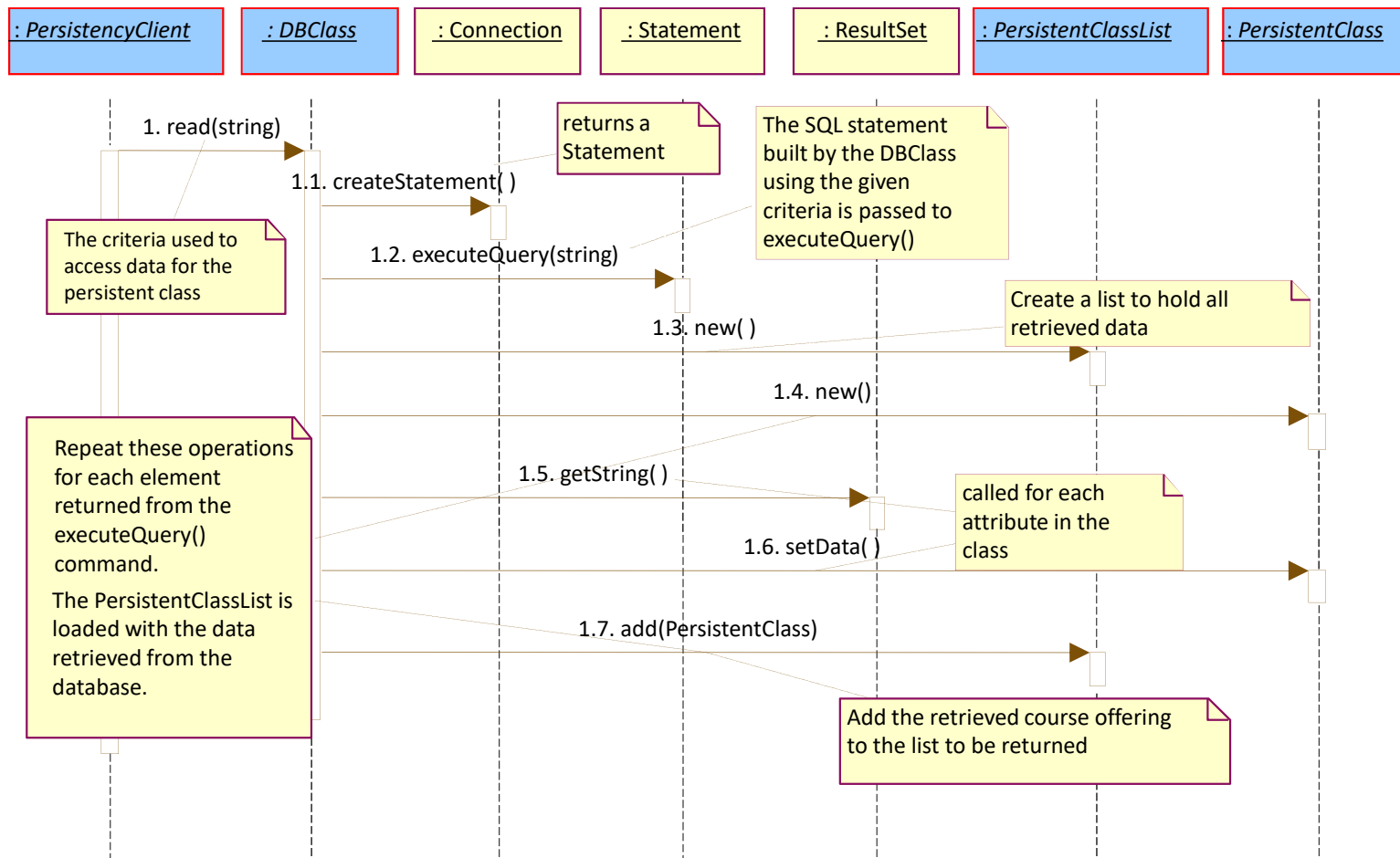
Initialize



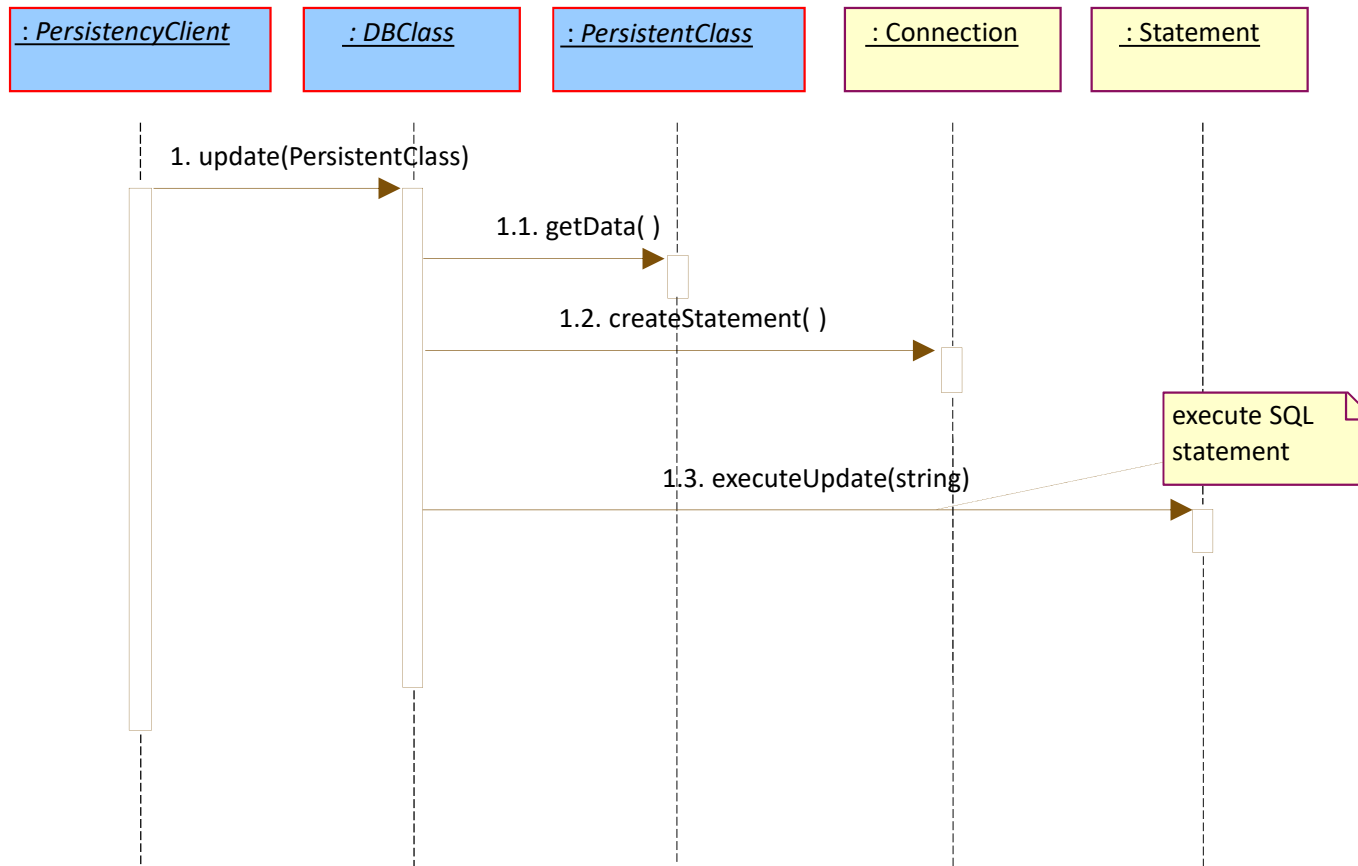
Create



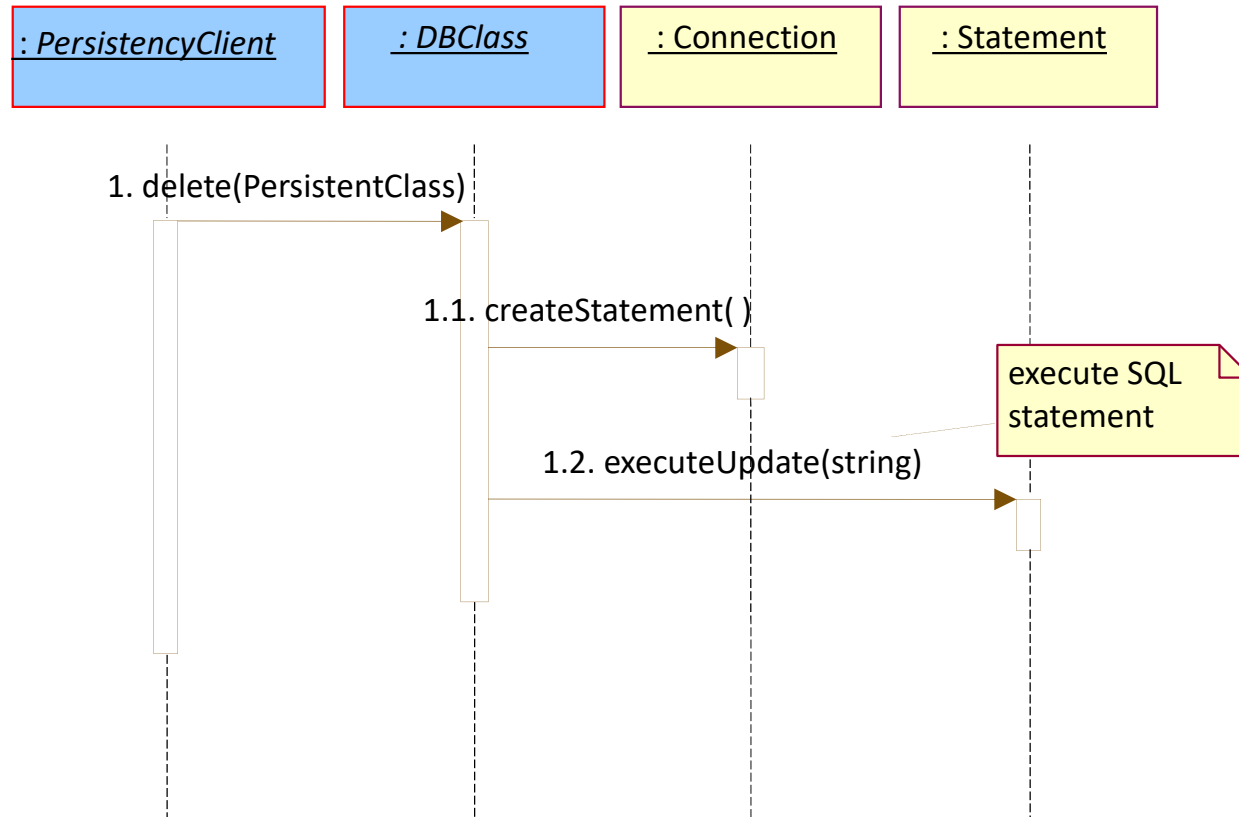
Read



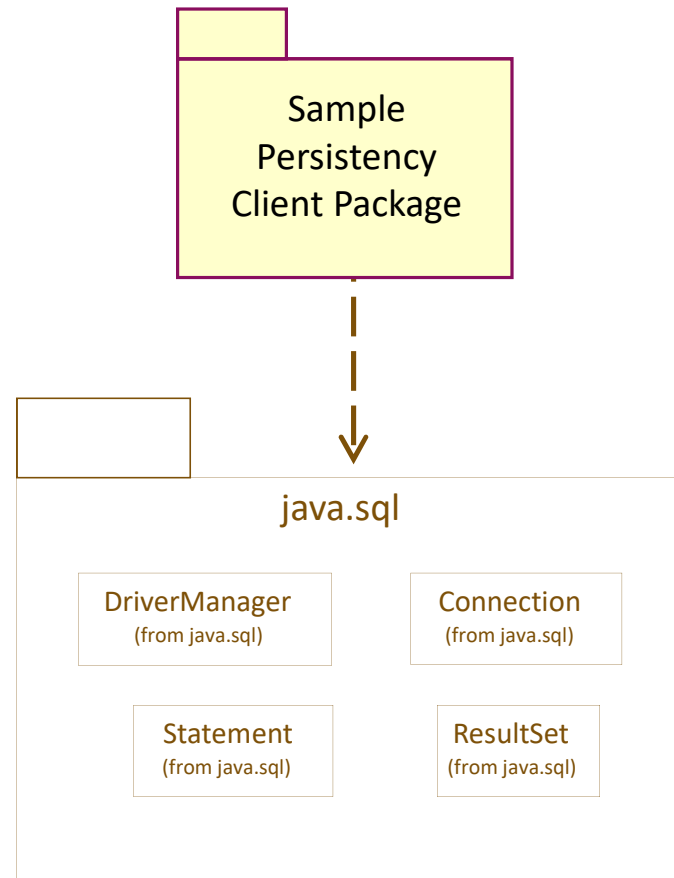
Update

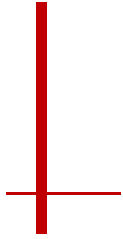


Delete



Incorporating JDBC





Xếp thời khóa biểu

- [\(PDF\) Giải thuật xếp thời khóa biểu ứng dụng vào bài toán quản lý xếp lịch thi kết thúc các lớp học phần tại Trường Đại học Cần Thơ \(researchgate.net\)](https://www.researchgate.net/publication/304862375_Giai_thuat_xep_thoi_khoa_bieu_ung_dung_vao_bai_toan_quan_ly_xep_lich_thi_ket_thuc_cac_lop_hoc_phan_tai_Truong_Dai_hoc_Can_Tho)
https://www.researchgate.net/publication/304862375_Giai_thuat_xep_thoi_khoa_bieu_ung_dung_vao_bai_toan_quan_ly_xep_lich_thi_ket_thuc_cac_lop_hoc_phan_tai_Truong_Dai_hoc_Can_Tho
- Áp dụng giải **thuật** di truyền vào bài **toán** sắp xếp **thời khóa biểu** cho toàn trường
- [I- Tìm hiểu về bài toán lập lịch \(hpu.edu.vn\)](https://lib.hpu.edu.vn/bitstream/handle/123456789/18245/27_HoangChinhNghia_CT901.pdf)
https://lib.hpu.edu.vn/bitstream/handle/123456789/18245/27_HoangChinhNghia_CT901.pdf
- [language agnostic - Algorithm for creating a school timetable - Stack Overflow](https://stackoverflow.com/questions/2177836/algorithm-for-creating-a-school-timetable)
<https://stackoverflow.com/questions/2177836/algorithm-for-creating-a-school-timetable>
- Genetic algorithm
- [pdf \(iop.org\)](https://iopscience.iop.org/article/10.1088/1742-6596/995/1/012050/pdf) <https://iopscience.iop.org/article/10.1088/1742-6596/995/1/012050/pdf>
- [Models and Algorithms for School Timetabling - A Constraint-Programming Approach \(d-nb.info\)](https://d-nb.info/967478146/34) <https://d-nb.info/967478146/34>
- timetable scheduling using genetic algorithm in python
- [FOC 523-528.pdf \(kdu.ac.lk\)](http://ir.kdu.ac.lk/bitstream/handle/345/3010/FOC%20523-528.pdf?sequence=1&isAllowed=y) <http://ir.kdu.ac.lk/bitstream/handle/345/3010/FOC%20523-528.pdf?sequence=1&isAllowed=y>
- [College Timetable using Time Scheduling Algorithm – IJERT](https://www.ijert.org/college-timetable-using-time-scheduling-algorithm) <https://www.ijert.org/college-timetable-using-time-scheduling-algorithm>