




HANOI UNIVERSITY OF
SCIENCE AND TECHNOLOGY



Bộ môn Công nghệ Phần mềm
Viện CNTT & TT
Trường Đại học Bách Khoa Hà Nội

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG


Bài 09. Tổng quan về UML và PTTK HĐT



Nội dung

1. Mô hình hóa
2. Tổng quan về UML
3. Phân tích thiết kế hướng đối tượng
4. Công cụ phát triển OOAD

2



Nội dung

1. **Mô hình hóa**
2. Tổng quan về UML
3. Phân tích thiết kế hướng đối tượng
4. Công cụ phát triển OOAD

3



1.1 Mô hình hóa là gì?

- Giúp đơn giản hóa thế giới thực bằng các mô hình
- Giúp hiểu rõ hơn về hệ thống dưới một góc nhìn nào đó

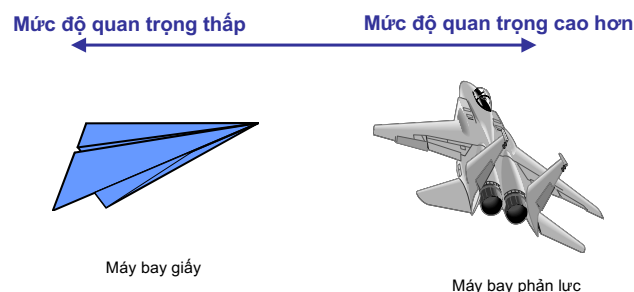






4

1.2. Sự quan trọng của mô hình hóa



1.2. Sự quan trọng của mô hình hóa (2)

- Rất nhiều đội dự án tiến hành xây dựng ứng dụng theo hướng tiếp cận của việc gấp máy bay giấy.
 - Bắt đầu lập trình ngay khi có được yêu cầu.
 - Mất rất nhiều thời gian và tạo ra rất nhiều mã nguồn.
 - Không có bất kỳ một kiến trúc nào.
 - Phải chịu khổ với những lỗi phát sinh.
- **Mô hình hóa là một con đường dẫn đến thành công của dự án.**

6

1.3. Vai trò của mô hình hóa hệ thống

- Hình dung một hệ thống theo thực tế hay theo mong muốn của chúng ta .
- Chỉ rõ cấu trúc hoặc ứng xử của hệ thống.
- Tạo một khuôn mẫu hướng dẫn nhà phát triển trong suốt quá trình xây dựng hệ thống.
- Ghi lại các quyết định của nhà phát triển để sử dụng sau này

7

1.4. Yêu cầu khi biểu diễn mô hình

- Chính xác (accurate): Mô tả đúng hệ thống cần xây dựng.
- Đồng nhất (consistent): Các view khác nhau không được mâu thuẫn với nhau.
- Có thể hiểu được (understandable): Cho những người xây dựng lẫn sử dụng
- Dễ thay đổi (changeable)
- Dễ dàng liên lạc với các mô hình khác.

8

Nội dung

1. Mô hình hóa
2. **Tổng quan về UML**
3. Phân tích thiết kế hướng đối tượng
4. Công cụ phát triển OOAD

9

2.1. UML là gì?

- Ngôn ngữ mô hình hóa thống nhất UML (Unified Modeling Language)

- UML là ngôn ngữ để:

- trực quan hóa (visualizing)
- xác định rõ (đặc tả - Specifying)
- xây dựng (constructing)
- tài liệu hóa (documenting)

các cấu phần (artifact) của một hệ thống phần mềm

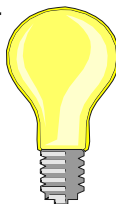


10

UML là ngôn ngữ trực quan

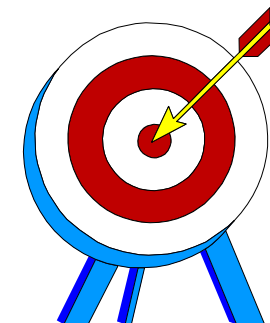
- UML là ngôn ngữ thống nhất trực quan giúp công việc được xử lý nhất quán, giảm thiểu lỗi xảy ra

- Có những thứ mà nếu không mô hình hóa thì không hoặc khó có thể hiểu được
- Mô hình trợ giúp hiệu quả trong việc liên lạc, trao đổi
 - Trong tổ chức
 - Bên ngoài tổ chức



UML là ngôn ngữ để đặc tả

- UML xây dựng các mô hình chính xác, rõ ràng và đầy đủ.

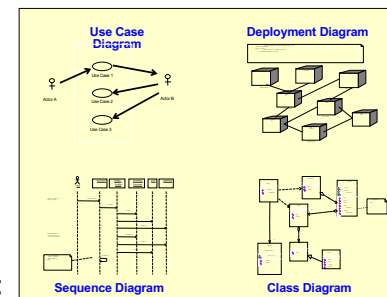


UML là ngôn ngữ để xây dựng HT

- Các mô hình UML có thể kết nối trực tiếp với rất nhiều ngôn ngữ lập trình.
 - Ánh xạ sang Java, C++, Visual Basic...
 - Các bảng trong RDBMS hoặc kho lưu trữ trong OODBMS
 - Cho phép các kỹ nghệ xuôi (chuyển UML thành mã nguồn)
 - Cho phép kỹ nghệ ngược (xây dựng mô hình hệ thống từ mã nguồn)

UML là ngôn ngữ để tài liệu hóa

- UML giúp tài liệu hóa về kiến trúc, yêu cầu, kiểm thử, lập kế hoạch dự án, và quản lý việc bàn giao phần mềm
- Các biểu đồ khác nhau, các ghi chú, ràng buộc được đặc tả trong tài liệu



2.2. Lịch sử phát triển của UML

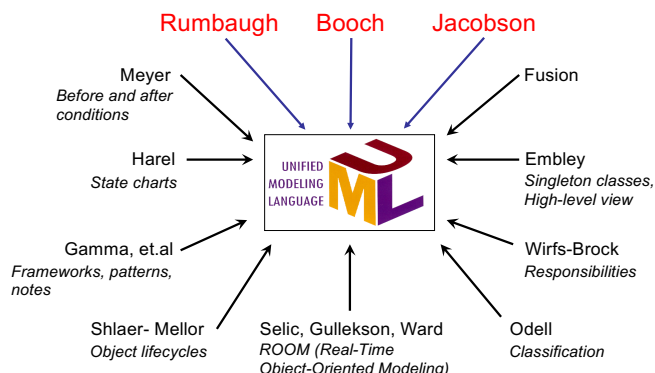
- Vào 1994, có hơn 50 phương pháp mô hình hóa hướng đối tượng:
 - Fusion, Shlaer-Mellor, ROOM, Class-Relation, Wirfs-Brock, Coad-Yourdon, MOSES, Syntropy, BOOM, OOSD, OSA, BON, Catalysis, COMMA, HOOD, Ooram, DOORS ...
 - "Meta-models" tương đồng với nhau
 - Các ký pháp đồ họa khác nhau
 - Quy trình khác nhau hoặc không rõ ràng
- Cần chuẩn hóa và thống nhất các phương pháp

2.2. Lịch sử phát triển của UML (2)

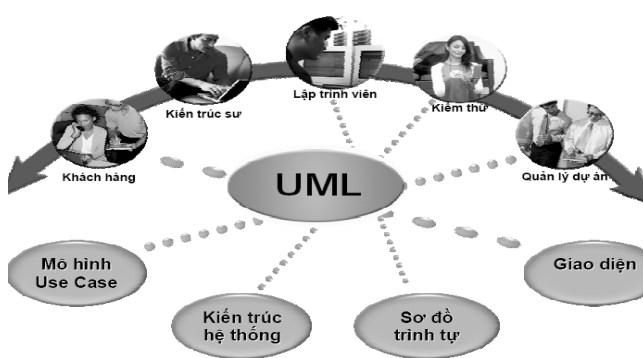
- UML được 3 chuyên gia hướng đối tượng hợp nhất các kỹ thuật của họ vào năm 1994:
 - Booch91 (Grady Booch): Conception, Architecture
 - OOSE (Ivar Jacobson): Use cases
 - OMT (Jim Rumbaugh): Analysis
- Thiết lập một phương thức thống nhất để xây dựng và "vẽ" ra các yêu cầu và thiết kế hướng đối tượng trong quá trình PTTK phần mềm → UML được công nhận là chuẩn chung vào năm 1997.



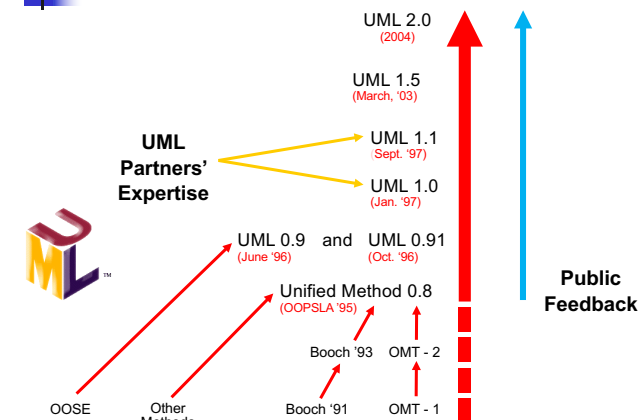
UML là một ngôn ngữ hợp nhất



UML là một ngôn ngữ thống nhất



2.2. Lịch sử phát triển của UML (3)

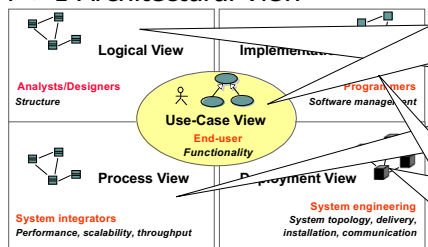


2.3. Các khung nhìn của UML

- Không đơn giản để mô hình hóa hệ thống phức tạp
- Lý tưởng: mô tả hệ thống trong 1 bản vẽ duy nhất → Bất khả thi
 - Một hệ thống cần được miêu tả trên các khía cạnh khác nhau: chức năng, phi chức năng, các thức hoạt động
- → Hệ thống phải được miêu tả theo các hướng nhìn khác nhau

2.3. Các khung nhìn của UML (2)

- Khung nhìn của mô hình có ý nghĩa với những người tham gia nào đó
- 4 + 1 Architectural View



Biểu diễn các chức năng và môi trường dự kiến của hệ thống dưới góc nhìn của người dùng

Mô tả các nút vật lý khác nhau và mô tả việc tổ chức các mô-đun phần mềm tính nhằm chia thành package, phân lớp và quản lý cấu hình

2.4. Các biểu đồ UML

- Biểu đồ:
 - là các hình vẽ bao gồm các ký hiệu phần tử mô hình hóa
 - minh họa một thành phần cụ thể hay một khía cạnh cụ thể của hệ thống.
- Một mô hình hệ thống thường có nhiều loại biểu đồ, mỗi loại có nhiều biểu đồ khác nhau.
- Một biểu đồ là một thành phần của một hướng nhìn cụ thể
- Một số loại biểu đồ có thể là thành phần của nhiều hướng nhìn khác nhau

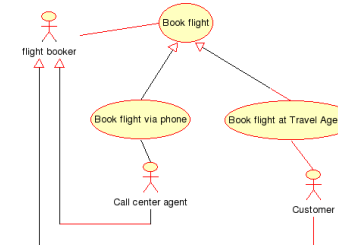
22

2.4. Các biểu đồ UML

- Biểu đồ use case (Use Case Diagram)
- Biểu đồ cấu trúc tĩnh (Static Structure Diagrams)
 - Biểu đồ lớp (Class Diagram)
 - Biểu đồ đối tượng (Object Diagram)
- Biểu đồ trạng thái (Statechart Diagram)
- Biểu đồ hoạt động (Activity Diagram)
- Biểu đồ tương tác (Interaction Diagrams)
 - Biểu đồ trình tự (Sequence Diagram)
 - Biểu đồ giao tiếp/cộng tác (Communication/Collaboration Diagram)
- Biểu đồ thực thi (Implementation Diagrams)
 - Biểu đồ thành phần (Component Diagram)
 - Biểu đồ triển khai (Deployment Diagram)

a. Biểu đồ use case

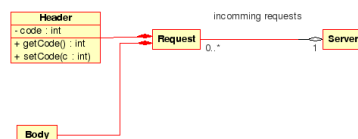
- Biểu đồ mô tả các yêu cầu chức năng của hệ thống dưới dạng các use case.
- Bao gồm các chức năng mong đợi của hệ thống (use case) và môi trường (actor) của nó.



24

b. Biểu đồ lớp (Class Diagram)

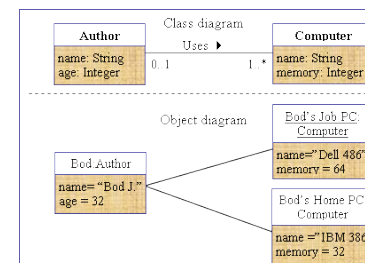
- Chỉ ra:
 - cấu trúc tĩnh của các lớp trong hệ thống
 - và các mối quan hệ giữa chúng



25

c. Biểu đồ đối tượng

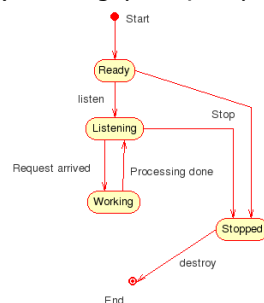
- Chỉ ra một loạt các đối tượng thực thể của lớp
- Là 1 ví dụ, bổ sung giải thích thêm cho biểu đồ lớp



26

d. Biểu đồ trạng thái (State Diagram)

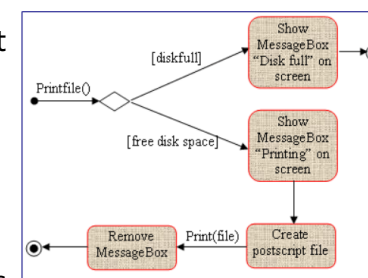
- Chỉ ra:
 - tất cả các trạng thái mà đối tượng của 1 lớp có thể có
 - và những sự kiện (event) nào sẽ gây ra sự thay đổi trạng thái
- Ví dụ các trạng thái của 1 server:



27

e. Biểu đồ hoạt động (Activity Diagram)

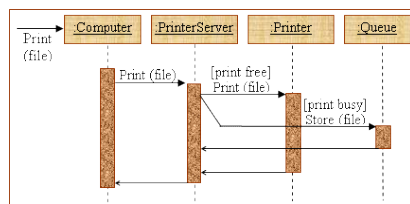
- Chỉ ra một trình tự lần lượt của các hoạt động trong 1 tiến trình xử lý
- Gồm các trạng thái hành động
 - Một trạng thái hành động sẽ qua đi khi hành động được thực hiện xong



28

f. Biểu đồ trình tự (Sequence Diagram)

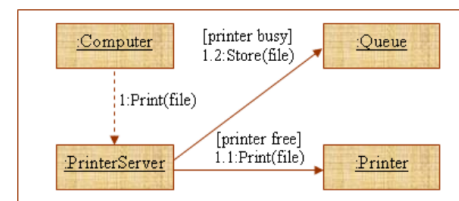
- Chỉ ra một cộng tác động giữa một loạt các đối tượng
- Nhấn mạnh trình tự & thời gian các thông điệp (message) được gửi giữa các đối tượng



29

g. Biểu đồ cộng tác (Collaboration Diagram)

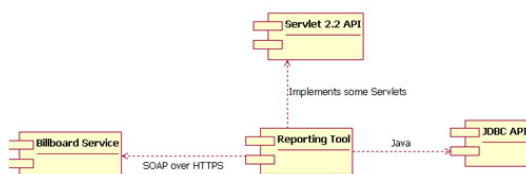
- Chỉ ra một sự cộng tác động
- Giống biểu đồ trình tự (Thường chỉ chọn 1 trong 2)
 - Ưu tiên ngữ cảnh: dùng biểu đồ cộng tác
 - Ưu tiên thời gian, trình tự: dùng biểu đồ trình tự



30

h. Biểu đồ thành phần (Component Diagram)

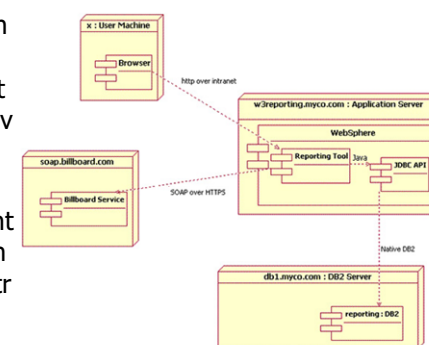
- Biểu diễn sự tổ chức và phụ thuộc giữa các thành phần phần mềm: mã nguồn, mã nhị phân (binary code) và những thành phần có khả năng thực thi



31

i. Biểu đồ triển khai (Deployment Diagram)

- Mô tả các tài nguyên vật lý trong hệ thống, bao gồm các nút (node), thành phần và kết nối.
- Mỗi mô hình chỉ bao gồm một deployment diagram hiển thị ánh xạ giữa những tiến trình xử lý tới thiết bị phần cứng

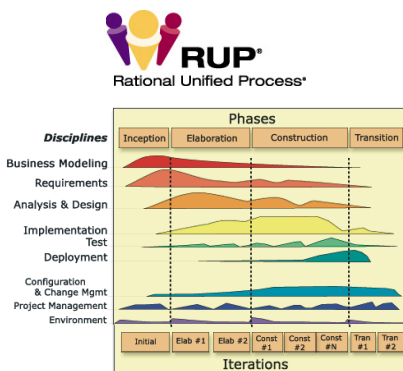


32

2.5. Quy trình và UML

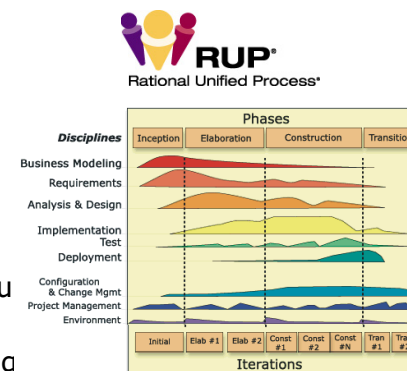
- UML là ký pháp chứ không phải là phương pháp

- UML có thể áp dụng cho tất cả các pha của quy trình phát triển phần mềm
- "Rational Unified Process" - quy trình phát triển cho UML



2.5. Quy trình và UML (2)

- RUP là quy trình công nghệ phần mềm phát triển bởi hãng Rational
- Là quy trình lặp và tăng trưởng từng bước
- Sử dụng các ký hiệu trực quan của UML
- Phát triển song song với UML



2.6. Ứng dụng của UML trong phân tích thiết kế hệ thống

- UML được sử dụng để phân tích nhiều loại hệ thống
 - Hệ thống thông tin (Information System)
 - Hệ thống kỹ thuật (Technical System)
 - Hệ thống nhúng (Embedded System)
 - Hệ thống phân tán (Distributed System)
 - Hệ thống nghiệp vụ (Business System)
 - Phần mềm hệ thống (System Software)

35

Nội dung

1. Mô hình hóa
2. Tổng quan về UML
3. **Phân tích thiết kế hướng đối tượng**
4. Công cụ phát triển OOAD

36

3.1. Tầm quan trọng của OOAD

- Nhiều người phát triển dự án
 - Cho rằng phần mềm chủ yếu được xây dựng bằng cách gõ "code" từ bàn phím
 - Không dành đủ thời gian cho quá trình phân tích và thiết kế phần mềm
- → Họ phải "cày bừa" để hoàn thành chương trình vì
 - Không hiểu hoặc hiểu sai yêu cầu
 - Giao tiếp với các thành viên không tốt
 - Không tích hợp được với module của đồng nghiệp...
- → Họ nhận ra rằng "Phân tích" và "Thiết kế" cần được coi trọng hơn, nhưng đã quá muộn

37

3.1. Tầm quan trọng của OOAD (2)

- Cần thiết lập một cơ chế hiệu quả để nắm bắt yêu cầu, phân tích thiết kế
- Cơ chế này phải như là một "ngôn ngữ thống nhất" giúp cho quá trình hợp tác hiệu quả giữa các thành viên trong nhóm phát triển phần mềm.
- → OOAD

38

3.2. Mục đích của OOAD

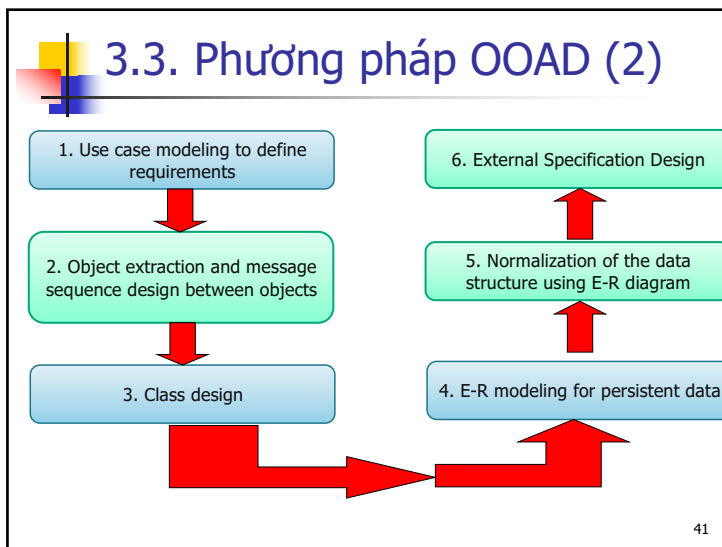
- Chuyển các yêu cầu của bài toán thành một bản thiết kế của hệ thống sẽ được xây dựng
- Tập trung vào quá trình phân tích các YÊU CẦU của hệ thống và thiết kế các MÔ HÌNH cho hệ thống đó trước giai đoạn lập trình
- Được thực hiện nhằm đảm bảo mục đích và yêu cầu của hệ thống được ghi lại một cách hợp lý trước khi hệ thống được xây dựng
- Cung cấp cho người dùng, khách hàng, kỹ sư phân tích, thiết kế nhiều cái nhìn khác nhau về cùng một hệ thống

39

3.3. Phương pháp OOAD

- OOAD được chia thành 2 giai đoạn
 - Phân tích hướng đối tượng (OOA)
 - Thiết kế hướng đối tượng (OOD)
- OOA là giai đoạn nhằm tạo ra các mô hình cơ bản (mô hình khái niệm) của hệ thống dựa theo những gì khách hàng yêu cầu về hệ thống của họ
- OOD sẽ bổ sung thêm các thông tin thiết kế chi tiết cho các mô hình nói trên

40



3.3.1. OOA

- Tạo được mô hình có các thành phần là đối tượng và khái niệm đời thực, dễ hiểu với người dùng
- Mô hình hóa các thực thể, giữ nguyên cấu trúc, quan hệ, hành vi giữa chúng

42

3.3.1. OOA (2)

- Ví dụ với 1 phòng bán ô tô:
 - Các thực thể:
 - Khách hàng
 - Người bán hàng
 - Phiếu đặt hàng
 - Phiếu (hoá đơn) thanh toán
 - Xe ô tô
 - Tương tác và quan hệ giữa các thực thể trên :
 - Người bán hàng dẫn khách hàng tham quan phòng trưng bày xe.
 - Khách hàng chọn một chiếc xe
 - Khách hàng viết phiếu đặt xe
 - Khách hàng trả tiền xe
 - Xe ô tô được giao đến cho khách hàng

43

3.3.2. OOD

- Tổ chức chương trình thành các tập hợp đối tượng cộng tác
 - Mỗi đối tượng là thực thể của một lớp
- Thiết kế trên kết quả của OOA
 - Cải thiện, tối ưu hóa thêm
 - Thiết kế các
 - Phương thức (operations)
 - Thuộc tính (attributes)
 - Mỗi quan hệ giữa các lớp (classes)
 - Đưa ra các biểu đồ tĩnh và động
 - Tĩnh: biểu thị các lớp và đối tượng
 - Động: biểu thị tương tác giữa các lớp & phương thức hoạt động

44

Thiết kế biểu đồ lớp

- Mục tiêu: cần xác định các thành viên của mỗi lớp và quan hệ giữa các lớp
- Một trong các kỹ thuật được ứng dụng nhiều nhất là Thẻ Class-Responsibility-Collaboration (CRC) card.
- Mỗi thẻ thể hiện một lớp, trên thẻ chúng ta lưu lại các thông tin sau về các lớp:
 - 1. Tên của lớp. Thông thường người ta đặt tên lớp liên quan đến vai trò của lớp, chúng ta sẽ sử dụng lớp để làm gì.
 - 2. Trách nhiệm của lớp: lớp có thể làm gì. Thông thường các thông tin ở đây bao gồm tên của các hàm thành phần
 - 3. Tương tác của lớp: lớp này có thể tương tác được với những lớp nào khác

45

Thiết kế đối tượng (1/2)

- Trong PT&TK hướng đối tượng người ta đã tổng kết 5 bước để thiết kế đối tượng:
 - Bước 1. Phát hiện đối tượng (Object discovery). Bước này được thực hiện ở giai đoạn phân tích chương trình.
 - Bước 2. Lắp ráp đối tượng (Object assembly). Bước tìm kiếm các đặc điểm của đối tượng để thêm vào các thuộc tính, các hàm thành phần cho đối tượng

46

Thiết kế đối tượng (2/2)

- Trong PT&TK hướng đối tượng người ta đã tổng kết 5 bước để thiết kế đối tượng:
 - Bước 3. Xây dựng hệ thống (System construction). Trong giai đoạn này chúng ta phát triển các đối tượng, xem xét các tương tác giữa các đối tượng để hình thành hệ thống hoạt động.
 - Bước 4. Mở rộng hệ thống (System extension). Khi chúng ta thêm vào các tính năng của hệ thống, cần thêm các lớp mới, các đối tượng mới và các tương tác giữa các đối tượng này với các đối tượng đã có trong hệ thống.
 - Bước 5. Tái sử dụng đối tượng (Object reuse). Đây là một trong những thử nghiệm quan trọng của các đối tượng và lớp trong thiết kế phần mềm. Chúng ta cần phải sử dụng lại các lớp và các đối tượng trong phần mềm (thông qua tính kế thừa và tương tác giữa các đối tượng)

47

Lưu ý (1/2)

- Một số điểm lưu ý khi phát triển các lớp
 - 1. Cần tạo ra lớp trước, sau đó mới nghĩ tới việc phát triển và hoàn thiện lớp trong quá trình giải quyết bài toán
 - 2. Khi phân tích hay phát triển các lớp không nên tập trung xác định tất cả thành viên một lớp, chúng ta sẽ biết rõ hơn khi phát triển hệ thống (learns as you go)
 - 3. Việc phát hiện ra các lớp cần thiết cho chương trình là một trong những nhiệm vụ chính của thiết kế hệ thống, nếu chúng ta đã có những lớp này (trong một thư viện lớp nào đó chẳng hạn) thì công việc sẽ dễ dàng hơn

48

Lưu ý (2/2)

- Một số điểm lưu ý khi phát triển các lớp
 - 4. Khi lập trình cần tuân thủ theo các thiết kế đã làm. Không nên bần khoăn khi không sử dụng phương pháp lập trình truyền thống và thấy choáng ngợp trước số lượng lớn các đối tượng.
 - 5. Luôn giữ nguyên tắc: mọi vấn đề cần giải quyết theo phương án đơn giản nhất, không phức tạp hóa. Sử dụng nguyên lý của Occam Razor: *Lớp đơn giản nhất bao giờ cũng là lớp tốt nhất, hãy bắt đầu bằng những cái đơn giản và chúng ta sẽ kết thúc bằng những hệ thống phức tạp*

49

Nội dung

1. Mô hình hóa
2. Tổng quan về UML
3. Phân tích thiết kế hướng đối tượng
4. **Công cụ phát triển OOAD**

50

4. Công cụ UML – OOAD

- Công cụ mã nguồn mở:
 - EclipseUML
 - UmlDesigner
 - ArgoUML...
- Công cụ thương mại:
 - Enterprise Architect
 - IBM Rational Software Architect
 - Microsoft Visio
 - Visual Paradigm for UML
 - SmartDraw...

51