Bài mẫu về sử dụng StarUML

để lập mô hình của một ứng dụng.

*“Hiền tài là nguyên khí của quốc gia”*

*Thân Nhân Trung*

*Người viết: Thân Văn Sử*

*Xin cám ơn giảng viên Lại Đức Hùng vì những đóng góp cho bài viết này.*

*Có rất nhiều tài liệu viết về UML- Unified Modeling Language- nhưng câu hỏi “Làm sao tôi có thể dùng UML để mô hình hoá ứng dụng?” hình như vẫn chưa có tài liệu nào giúp người mới học tiếp cận một cách dễ dàng.*

*Vẫn biết rằng mô hình hoá bằng UML là qúa trình có tính xoắn ốc và tinh chỉnh dần (lặp), nhiều quan điểm về mô hình hoá cũng được giới thiệu. Có lẽ vì đặc trưng lặp và năng cao dần nên hình như có rất ít bài viết hướng dẫn từng bước cho người mới tiếp cận và người mới tiếp cận luôn mong mỏi câu trả lời cho câu hỏi “Tôi sẽ bắt đầu thế nào?”.*

*Bài viết này chỉ nhằm hướng dẫn từng bước giúp những người mới làm quen với UML biết các bước thực hiện việc dùng một phần mềm UML nào đó có thể lên mô hình cho bài toán ở mức cơ bản nhất thông qua một bài toán cụ thể cũng như cung cấp những ý tưởng để viết báo cáo đi từ phân tích yêu cầu đến kết quả với một mong muốn duy nhất là chỉ ra một cách bắt đầu cho người mới tiếp cận với kỹ thuật này. Sau khi thực hiện xong bài mình hoạ này, người viết mong rằng các bạn có thể tự mô hình hoá lấy bài toán của mình. Mong thay.*

*Phần mềm được dùng để minh hoạ là StarUML 5.0.2.1570. Tuy nhiên, nếu các bạn dùng công cụ khác thì mọi thứ cũng tương tự.*

*Trong bài viết này, phần chữ mầu xanh và chữ nghiêng nhằm giải thích và không đưa vào báo cáo phân tích cũng như thiết kế.*

*Với kiến thức và kỹ năng hữu hạn, chắc chắn bài viết còn nhiều thiếu sót hoặc sai lầm, người viết xin chân thành nhận những góp ý từ tất cả mọi người và xin để những thiếu sót này để người đọc tự hoàn thiện dần trong các bước lặp tăng dần theo đặc điểm của phương pháp nàyt*

*01-05-2011*

*Thân Văn Sử*

*Emails:*

[*sutv@fpt.com.vn*](mailto:sutv@fpt.com.vn)

*sutv@fpt.edu.vn*

**Bài toán**

Nhà sách Dục Trí muốn xây dựng một phần mềm quản lý và bán lẻ sách cho khách hàng với mô tả sau:

Mỗi quyển sách chứa các thông tin: Tựa sách, tác giả, nhà xuất bản, giá mua, giá bán, số tồn.

Sách của nhà sách được sắp xếp theo thể loại và được trưng bầy cho khách hàng xem tại các kệ. Một thể loại sách chỉ được trưng bầy ở một kệ và một kệ trưng bày nhiều thể loại sách.

Kế toán của nhà sách (1 người) chịu trách nhiệm về danh mục sách được bán trong nhà sách.

Nhân viên thu ngân- thường là 3 người- sẽ in hoá đơn cho khách hàng khi khách hàng mang sách muốn mua từ kệ sách ra quầy thu ngân.

Người quản lý (1 người) sẽ xem xét tình hình kinh doanh và chịu trách nhiệm về nhân sự.

Ngoài các chức năng quản lý, người quản lý muốn cung cấp cho khách hàng một chương trình giúp khách hàng tìm kiếm sách khi họ đã biết vài thông tin như: tựa sách, tên tác giả, nhà xuất bản

1. **Phân tích hệ thống**

*(Bước này giúp nhận diện các actor, và dựa trên trách nhiệm của từng người tham gia vận hành hệ thống, các usecase được nhận diện. Từ các khái niệm liên quan đến hệ thống giúp xây dựng từ điển dữ liệu. Từ từ điển dữ liệu giúp nhận dạng dữ liệu cần lưu trữ để xây dựng database)*

* 1. **Nhận diện actors và use case**

***Gọi ý về nhận diện các actor và usecase***

* *Từ bản mô tả tài toán, nhặt ra những người tham gia hệ thống (****danh từ****) hay những hệ thống khác- như máy quẹt thẻ, máy quẹt vân tay,… đó chính là các actor*
* *Trách nhiệm của người, chức năng của máy là những usecase. Các trách nhiệm, chức năng này trong mô tả bài toán được diễn đạt bằng các* ***động từ vì con người dùng động từ để mô tả hành động***
* *Khi phân tích usercase, dùng kỹ thuật phân tích top-down đi từ các tác vụ mức cao xuống các usecase mức thấp hơn cho đến các usecase nguyên tử (usecase không thể phân tích thấp hơn nữa, một công việc thấp nhất, nhỏ nhất còn có thể mô tả bằng một động từ). Điều này cũng giống như con người khi cụ thể hoá các trách nhiệm phải làm cho từng người trong một tổ chức. Việc phân tích đến acc1 usecase nguyên tử sẽ tạo ra các thuận lợi cho việc mô hình các view khác (sequence diagram, class diagram, State chart..), cho việc kiểm tra tính hợp lệ giữa các view, cho các công việc liên quan với usecase như sinh code, viết testcase, test report.*
* *Các usecases có thể được chia về một số các lược đồ tương tự như chia từng mức để các lược đồ đủ đơn giản và dễ hiểu.*
* *Trong bài viết này các lược đồ có thể có cả những hình thừa vì để giúp các bạn thực hiện. Các bạn không nên đưa các hình thừa vào báo cáo thiết kế.*

Từ mô tả của bài toán, bốn actors được nhận diện ứng với bốn vai trò của người dùng: người quản lý, kế toán, nhân viên, khách hàng. Ba actor quản lý, kế toán, nhân viên là những actor cần đăng nhập vào hệ thống nên được khái quát hoá thành actor CanBo. Những người dùng ứng với bốn vai trò này sẽ tham gia vận hành hệ thống cùng với các usecase sau:

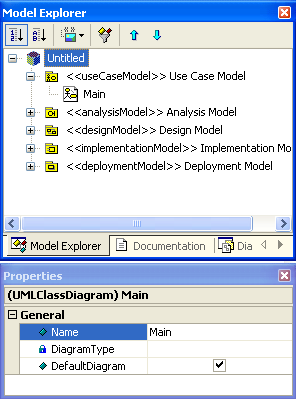
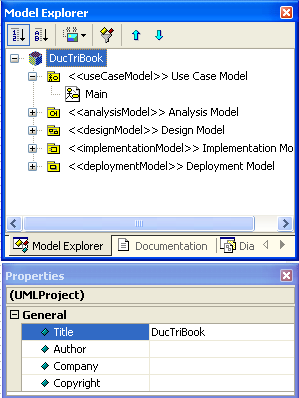
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Người | Số lượng | Thao tác với hệ thống | Use case |
| 1 | CanBo | Nhiều | LogIn | UC\_LogIn |
| 2 | Quản lý | 1 | Thống kê và quản lý nhân sự   * Thống kê kinh doanh * Quản lý nhân sự * Thêm nhân viên * Sửa nhân viên * Cho nhân viên nghỉ việc | UC\_TK\_NS   * UC\_ThongKe * UC\_QLNhanSu * UC\_ThemNV * UC\_SuaNV * UC\_XoaNV |
| 3 | Kế toán | 1 | Quản lý nhà sách   * Quản lý nhà xuất bản * Thêm nhà xuất bản * Sửa nhà xuất bản * Xoá nhà xuất bản * Quản lý Tác giả * Quản lý kệ sách * Quản lý sách * Quản lý mua sách | UC\_QLNhaSach   * UC\_QLNhaXuatBan * UC\_ThemNXB * UC\_SuaNXB * UC\_XoaNXB * UC\_QLTacGia * UC\_QLKeSach * UC\_QLSach * UC\_QLMuaSach |
| 4 | Nhân viên | 3 | In hoá đơn | UC\_InHoaDon |
| 5 | Khách hàng | Nhiều | Xem và tìm sách   * Xem danh mục sách * Tìm sách | UC\_Xem\_Tim\_Sach   * UC\_XemSach * UC\_TimSach |

***Trong bảng trên, chúng tôi mới chỉ phân tích kỹ đến UC\_QLNhaXuatBan. Các usecase còn lại để dành cho các bạn tự phân tích.***

***Hướng dẫn cách tạo một project với StarUML***

* *Khởi động StarUML*
* *File/New Project.*
* *File/Save as*
* *Đặt tên DucTriBook.*

*Một cách để đổi tên đối tượng là chọn đối tượng, vào cửa sổ* ***Properies*** *( menu View/Properties),đặt tên mới cho đối tượng ở dòng* ***Title/Name***

*Vì nhắm đến xây dựng ứng dụng bằng ngôn ngữ Java đồng thời lấy được các kiểu dữ liệu cơ bản của Java, chúng ta xác định profile là Java – xem ở cuối tài liệu này:*

***Hướng dẫn cách tạo một Usecase Diagram***

*Trong Model Explorer, chọn Use Case Model, chọn Main, đổi tên thành Ucases\_0 ( dùng của sổ Properties) chọn các chức năng trong Tookbox bên trái để vẽ sơ đồ này.*

***Hướng dẫn thao tác với Toolbox:***

* *NhómUsecase chứa các chức năng giúp vẽ usecase*
* *Nhóm Annotaion chứa các chức năng vẽ ô chứa giải thích*

***Hướng dẫn thao tác với Usecase Toolbox***

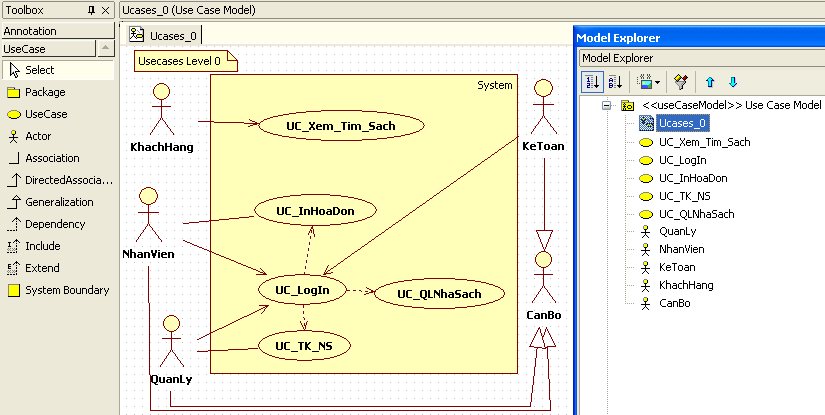
* *Chọn chức năng* ***Select*** *rồi kéo thả các actors đến các vị trí thích hợp*
* *Chọn chức năng* ***UseCase*** *thêm các usecase với tên như hình sau.*
* *Chọn chức năng* ***System Boundary*** *để vẽ biên của hệ thống*
* *Chọn chức năng* ***Association/Directed Association/Dependency/Include*** *để tạo sự kết hợp giữa các đối tượng trong sơ đồ.*

***Gợi ý về sự kết hợp giữa các đối tượng***

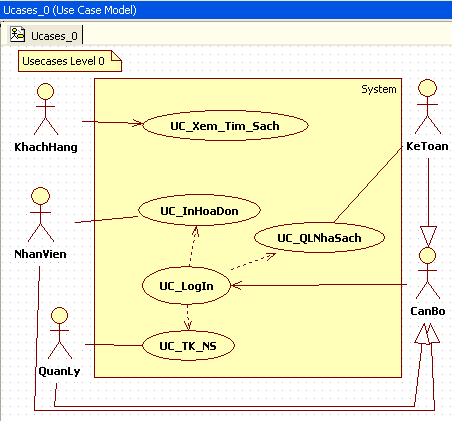
* *Kết hợp dạng phụ thuộc (Dependency):*

*Ucase\_Yếu ----> Ucase\_Mạnh. Khi usecase mạnh thay đổi, usecase yếu có thể bị thay đổi theo.*

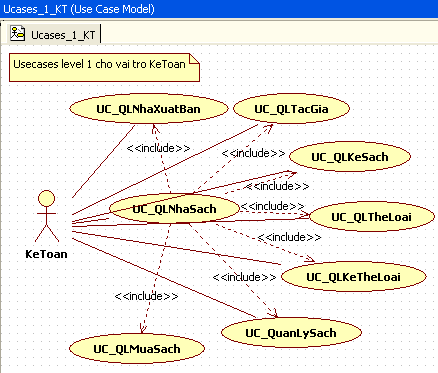
* *Có những kết hợp trực tiếp giữa các thành phần: Mũi tên nét liền (DirectedAssociation).*
* *Có những kết hợp gián tiếp hoặc phải thông qua một kết hợp khác (đoạn thẳng – Association)*
* *Quan hệ include: Ucase này chứa các hành vi được khai báo trong khai báo khác: Including\_Ucase -----> Included\_Ucase.*
* *Quan hệ extend: Usecase này là mở rộng (thêm tính chất🡪 chi tiết hơn🡪 usecase con) từ usecase khác( ít chi tiết hơn, usecase cha). Cách diễn đạt: Ucase\_con -----> Ucase\_cha*



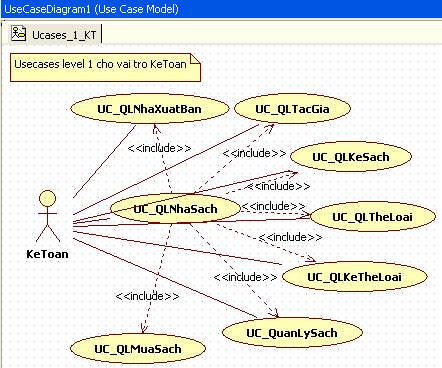
*Hình vẽ này không cho thấy role CanBo tương tác với hệ thống, chúng ta nên vẽ lại.*



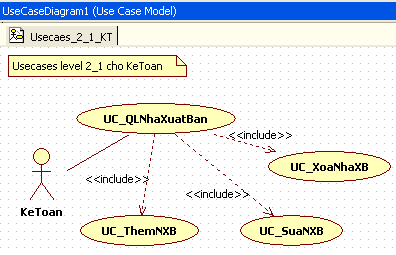
*Hình 1: Các usecase mức 0*



*Hình 2: Các usecase mức 1 cho vai trò QuanLy*



*Hình 3: Các usecase mức 1 cho vai trò KeToan*



*Hình 4: Các usecase mức 2\_1 cho vai trò KeToan*

***(Các usescase diagram còn lại xin dành để các bạn tự xây dựng)***

* 1. **Từ điển dữ liệu**

Từ mô tả của bài toán, các khái niệm có trong bài toán được nhận diện và sử dụng trong bản phân tích bao gồm:

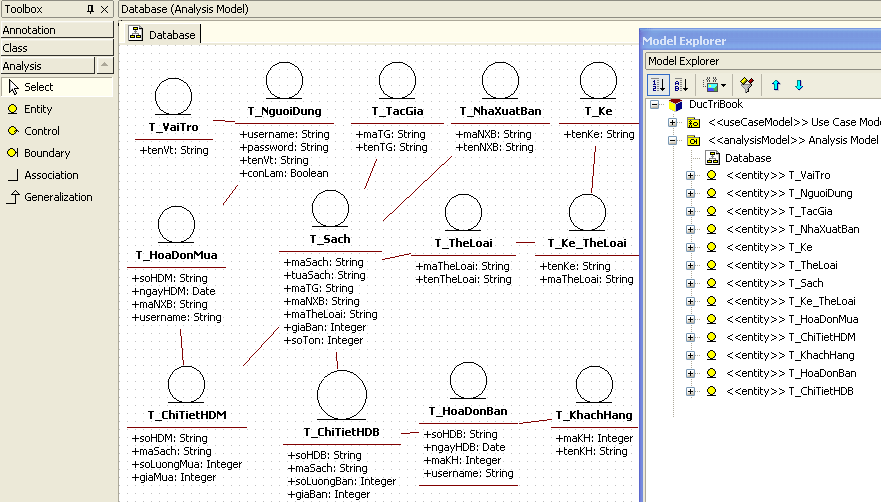
|  |  |
| --- | --- |
| **Tên** | **Mô tả** |
| HoaDonBan | Hoá đơn bán |
| HoaDonNhap | Hoá đơn nhập sách |
| Ke | Kệ sách |
| KhachHang | Khách hàng |
| NguoiDung | Quản lý/ kế toán, nhân viên/Khách hàng |
| NhaXuatBan | Nhà xuất bản |
| Sach | Sách |
| TacGia | Tác giả |
| VaiTro | Vai trò của người tham gia hệ thống |

* 1. **Phân tích kho dữ liệu**

Từ từ điển dữ liệu, kho dữ liệu của ứng dụng được mô hình hoá bằng các bảng sau:

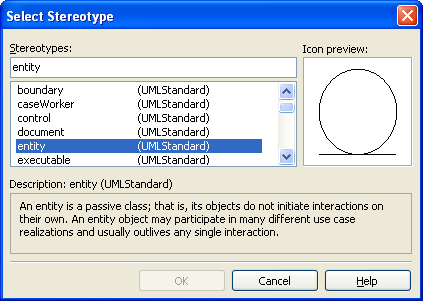
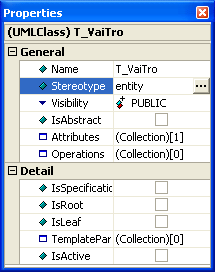
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stt** | **Tên thực thề** | **Thuộc tính** | **Kiểu** | **Mô tả cho** |
| 1 | T\_VaiTro | tenVt | String(20) - PK | Vai trò của người chạy chương trình  Trị: Quản lý/ Kế toán/Nhân viên |
| 2 | T\_NguoiDung | username  password  tenVt  conLam | String(20)- PK  String(20)  String (20) – FK  Boolean | Người dùng có authentication   * Tên người dùng * Mật khẩu * Vai trò của người dùng * Còn làm hay nghỉ rồi |
| 3 | T\_NhaXuatBan | maNXB  tenNXB | String(20)-PK  String(20) | Nhà xuất bản   * Mã nhà xuất bản * Tên nhà xuất bản |
| 4 | T\_TacGia | maTG  tenTG | String(20)-PK  String(50) | Tác giả   * Mã tác giả * Tên tác giả |
| 5 | T\_Ke | tenKe | String(20) - PK | Kệ sách  -tên kệ sách |
| 6 | T\_TheLoai | maTheLoai  tenTheLoai | String(20)-PK  String(50) | Thể loại sách   * Mã thể loại * Tên thể loại |
| 7 | T\_Ke\_TheLoai | tenKe  maTheLoai | String(20) – PK, FK  String(20) – PK, FK | Quan hệ kệ(n)-Thể loại(n)   * Tên kệ chứa thể loại * Mã thể loại được chứa trên kệ |
| 8 | T\_Sach | maSach  tuaSach  maTG  maNXB  maTheLoai  giaBan  soTon | String(20)-PK  String (30)  String (20) –FK  String (20) - FK  String (20) - FK  Integer  Integer | Sách   * Mã sách * Tựa sách * Mã tác giả * Mã nhà xuất bản * Mã thể loại * Giá bán * Lượng tồn kho |
| 9 | T\_KhachHang | maKH  tenKH | Integer, PK, identity  String(50) | Khách hàng   * Mã khách hàng * Tên khách hàng |
| 10 | T\_HoaDonMua | soHDM  ngayHDM  maNXB  username | String(20) - PK  Date  String(20) –FK  String(20) –FK | Hoá đơn mua   * Số hoá đơn mua * Ngày hoá đơn * Mã nhà xuất bản xuất hoá đơn * Kế toán lưu hoá đơn |
| 11 | CTHDM | soHDM  maSach  soluongMua  giaMua | String(20) –PK, FK  String(20)- PK, FK  Integer  Integer | Chi tiết hoá đơn mua   * Số hoá đơn mua * Mã sách được mua * Số lượng mua * Giá mua |
| 12 | T\_HoaDonBan | soHDB  ngayPD  maKhach  username | String(20)- PK  Date  Integer  String(20) - FK | Hoá đơn bán sách   * Số hoá đơn bán * Ngày hoá đơn bán * Mã khách hàng * Nhân viên in hoá đơn |
| 13 | T\_CTHDB | soHDB  maSach  soluongBan  giaBan | String(20) - FK, PK  String(20) – FK, PF  Integer  Integer | Chi tiết hoá đơn bán   * Số hoá đơn bán * Mã sách đã bán * Số lượng * Đơn giá bán |

Các thực thể này đóng vai trò các kho dữ liệu để người dùng thao tác vào. Mô hình hoá các thực thể được biều diễn bằng mô hình sau: *(trong báo cáo chỉ để diagram, không có các cửa sổ công cụ)*



***Hướng dẫn cách tiến hành***

* *Chọn Main diagram trong Analysis Model, đổi tên thành Database ( dùng cửa sổ Properties)*
* *Lần lượt chọn biểu tượng Entity trong Toolbox để vẽ các entity*
  + *Bảo đảm mô hình biểu diễn đúng entity: Chọn đối tượng trong Model Explorer, mở cửa sổ Properties ( nếu không thấy thì dùng menu View.Properties), chọn stereoType: Entity.*

**

* + *Thay đổi cách trình bầy 1 entity trên lược đồ: Kích phải vào 1 entity trên lược đồ, chọn* ***Format/Stereotype display/Iconic*** *kết hợp với chọn* ***Format/Suppress Asttributes****. Stereotype- kiểu có thể thay đổi, lưỡng tính- là một kiểu dữ liệu của UML cho phép người thiết kế cụ thể hoá sau một đối tượng.*
  + *Vì đây là mô hình dữ liệu nên dùng những kiến thức về database để phân tích.*
  + *Hai cách add attribute cho entity:*
    - *Cách 1: Kích phải chuột vào entity trên diagram, add attribute, gõ trực tiếp tên và kiểu của attribute phân cách bằng dấu hai chấm.*
    - *Cách 2: Kích phải vào entity trong của sổ Model Explorer/ Add attribute/ ấn định tên và kiểu trong cửa sổ Properties.*
  + *Phân tich này sẽ được Database specialist thiết kế database cho ứng dụng*
  + *Không nên tạo operation cho các entity vì dữ liệu trong database và đối tượng trong bộ nhớ có khác biệt một chút. Chúng ta sẽ tạo những class ánh xạ riêng cho các bảng trong đó có các operation giúp nạp/ lưu mới/ cập nhật/ xoá giữa đối tượng thuộc lớp với database bên dưới.*
  1. **Chi tiết hoá các use case**

Các usecase đã phân tích ở trên được chi tiết hoá bằng các mô tả sau:

**1.4.1- Nhóm use case chung**

|  |  |  |
| --- | --- | --- |
| User Case ID | UC\_1 | |
| Name | UC\_LogIn | |
| Goal | Đăng nhập vào hệ thống | |
| Actors | CanBo | |
| Pre-conditions |  | |
| Post-conditions |  | |
| Main Flow | 1.Người dùng nhập username, password, tenVt | 1. Hệ thống sẽ kiểm tra tính hợp lệ của user.   Xuất giao diện phù hợp cho user này. |
| Exception | 2A: Nếu user không hợp lệ, xuất thông báo “Dữ liệu không hợp lệ” và quay về bước 1. | |
| Open Issues | N/A | |

**1.4.2- Nhóm use case cho vai trò quản lý**

|  |  |  |
| --- | --- | --- |
| User Case ID | UC\_2 | |
| Name | UC\_TK\_NS | |
| Goal | Kết hợp xem tình hình kinh doanh và quản lý nhân sự | |
| Actors | QuanLy | |
| Pre-conditions | User đã đănng nhập thành công với vai trò QuanLy | |
| Post-conditions |  | |
| Main Flow | 1.User chọn chức năng | 1. Nếu chức năng được chọn là “Thống kê”, xuất giao diện của UC\_ThongKe   Ngược lại xuất giao diện của UC\_QLNhanSu. |
| Exception |  | |
| Open Issues | N/A | |

|  |  |  |
| --- | --- | --- |
| User Case ID | UC\_3 | |
| Name | UC\_ThongKe | |
| Goal | Thống kê tình hình kinh doang trong một khoảng thời gian | |
| Actors | QuanLy | |
| Pre-conditions | User đã đănng nhập thành công với vai trò QuanLy | |
| Post-conditions |  | |
| Main Flow | 1.User nhập date1, date2 | 1. Xuât thống kê kinh doanh từ date1 đến date2. |
| Exception | 2A: Dạng ngày tháng không hợp lệ: Báo lỗi “Invalid date forma” quay lại bước 1  2B: Nếu date1>date2, hoán vị date1 với date2, tiếp tục bước 2 | |
| Open Issues | N/A | |

|  |  |  |
| --- | --- | --- |
| User Case ID | UC\_4 | |
| Name | UC\_QLNhanSu | |
| Goal | Quản lý nhân sự của nhà sách | |
| Actors | QuanLy | |
| Pre-conditions | User đã đănng nhập thành công với vai trò QuanLy | |
| Post-conditions |  | |
| Main Flow | 2.User chọn chức năng | 1. Hệ thống xuất danh sách nhân viên  3. Hệ thống cho phép user thực hiện chức năng. |
| Exception |  | |
| Open Issues | N/A | |

|  |  |  |
| --- | --- | --- |
| User Case ID | UC\_5 | |
| Name | UC\_ThemNV | |
| Goal | Thêm nhân viên mới | |
| Actors | QuanLy | |
| Pre-conditions | User đã đănng nhập thành công với vai trò QuanLy | |
| Post-conditions |  | |
| Main Flow | 1. User nhập username, password, vai trò của nhân viên mới. | 2. Hệ thống lưu thông tin nhân viên mới vào kho dữ liệu. |
| Exception | 2A: Nếu username trống, báo lỗi “username khải không trống” và quay lại bước 1  2B: Nếu password dưới 6 ký tự, báo lỗi “Password: ít nhất 6 ký tự” và quay lại bước 1  2C: Nếu username trùng với các username đang có, báo lỗi “Trùng tên” và quay lại bước 1 | |
| Open Issues | N/A | |

|  |  |  |
| --- | --- | --- |
| User Case ID | UC\_6 | |
| Name | UC\_SuaNV | |
| Goal | Sửa thông tin nhân viên | |
| Actors | QuanLy | |
| Pre-conditions | User đã đănng nhập thành công với vai trò QuanLy | |
| Post-conditions |  | |
| Main Flow | 1. User nhập username của nhân viên cần sửa  3. User nhập password, vai trò mới của nhân viên hiện hành. | 2. Hệ thống xuất thông tin của nhân viên đã được chọn.  4. Hệ thống cập nhật thông tin về nhân viên này. |
| Exception | 2A: Nếu username không tồn tại, báo lỗi :User không tồn tại”, quay lại bước 1  2B: Nếu username lại là chính user QuanLy, báo lỗi “Không thề xoá chính mình”, về bước 1  4A: Nếu password dưới 6 ký tự, báo lỗi “Password: ít nhất 6 ký tự” và quay lại bước 3  4B: Nếu vai trò không trùng với các vai trò đang có, báo lỗi “Vai trò không hợp lệ” và quay lại bước 3 | |
| Open Issues | N/A | |

**1.4.3- Nhóm use case cho vai trò kế toán**

***Bạn tự làm***

**1.4.4- Nhóm use case cho nhân viên**

***Bạn tự làm***

**1.4.5- Nhóm use case cho khách hàng**

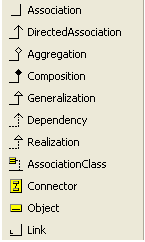
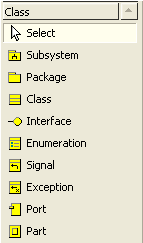
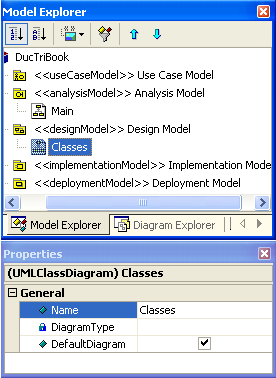
***Bạn tự làm***

1. **Thiết kế**
   1. **Sơ đồ lớp cần có của hệ thống**

***Phần giải thích***

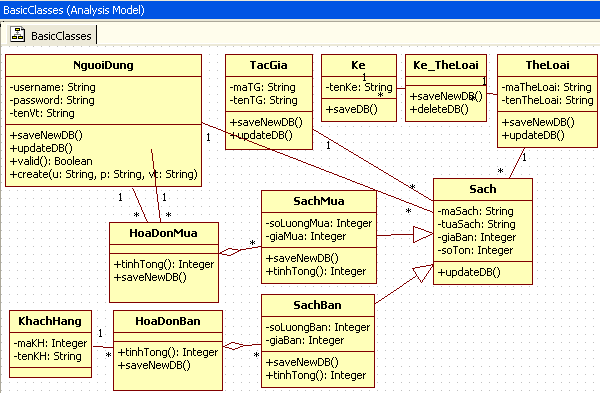
*Sơ đồ lớp là sơ đồ tĩnh vì sơ đồ này chỉ cho biết các bản mô tả về các đối tượng có trong hệ thống. Các lớp không khó được nhận diện. Gợi ý sau đây giúp nhận diện các lớp và tạo Class Diagrams:*

* *Các lớp ánh xạ cho các bảng trong database- thường thì một bảng sẽ ứng với một lớp.*
* *Các lớp tạo tiện ích như các danh sách (chứa danh sách lấy từ database ra hoặc chứa danh sách trước khi lưu vào database).*
* *Các lớp đóng vai trò giao diện người dùng (GUI)*
* *Một số chú ý:*
  + *Class Diagrams có thể trong Analysis Model hoặc Design Model đều được vì chỉ trong hai nhóm này mới cho ta tạo class diagram.*
  + *Khai thác tính thừa kế để đơn giản hoá các lớp*
  + *Khi tạo 1 class, chỉ khai báo các dữ liệu và hành vi cần thiết. Những dữ liệu giúp mô tả quan hệ giữa các lớp cũng không cần mô tả mà dùng chức năng Association. Những hành vi cơ bản như constructors, getters, setters không cần thiết kế vì khi StarUML sinh code****- chỉ sinh ra khung của lớp – các hành vi được sinh ra đều code trống-***  *các construtor được xem như hành vi* ***bình thường (có kiểu trả trị) trong khi đó các môi trường lập trình như NetBeans đã hỗ trợ đầy đủ các đặc điểm này. Khi sinh code, các quan hệ sẽ được mô tả tự động bằng biến tham khảo đến lớp mà lớp này có quan hệ. Việc sinh code được đề cập sau.***
  + *Các công cụ giúp thao tácvới class Diagram:*



Từ các phân tích về dữ liệu và các usecase ở trên, các lớp sau được nhận diện:

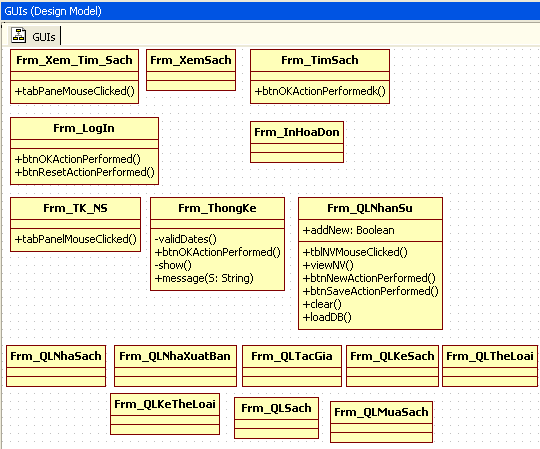
**2.1.1- Các lớp ánh xạ với kho dữ liệu**



*Các bạn làm rõ thêm các hành vi cho các lớp. Thường một lớp ánh xạ với bảng thường có các hành vi: nạp, tạo mới, sửa, xoá, đọc một danh sách các dòng (có thể theo một điều kiện nào đó phù hợp với bài toán).*

**2.1.2- Các lớp tạo giao diện người**

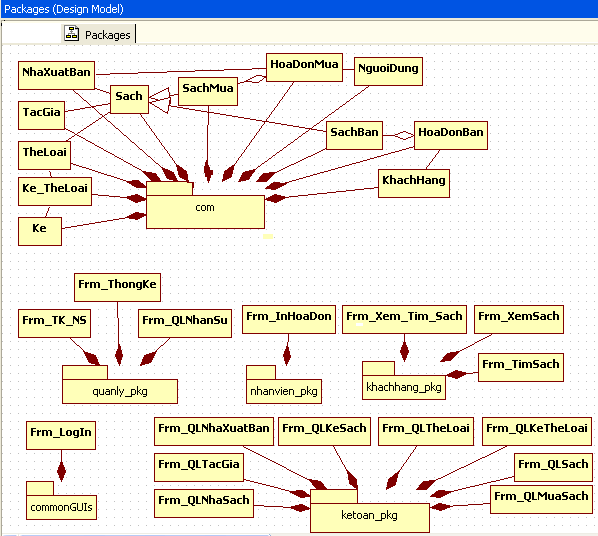
Mỗi chức năng của chương trình (usecase) cần có một giao diện người dùng ( GUI-các class). Từ phân tích usecase ở trên, các giao diện người dùng được cần đến ứng với nhu cầu tương tác của từng actor:



***Hướng dẫn cách thực hiện***

* *Trong* ***Model Explorer****, kích phải vào* ***Design Model****, chọn* ***Add Diagram/Class diagram***
* *Mở cửa sổ* ***Properties*** *đổi tên class diagram này là* ***Interfaces***
* *Dùng chức năng* ***class*** *trong* ***Class Toolbox*** *để vẽ các lớp GUI.*
* *Nếu ứng dụng được nhắm đến là ứng dụng chạy trên mạng, các interface (bộ hành vi giao tiếp lập trình) được cần đến, chúng ta cũng làm tương tự.*
* *Để thay đổi định dạng các đối tượng trên sơ đồ: chọn 1 đối tượng ( hoặc dùng chuột kéo đánh dấu một số đối tượng, hoặc Ctrl A để đánh dấu mọi đối tượng), vào menu format, chọn* ***Suppress Attributes/ Suppress Operations***.
  1. **Thiết kế gói**

Vì user được chia thành 5 vai trò và để có những thuận lợi trong việc bảo trì hệ thống, 6 gói sau được chọn:



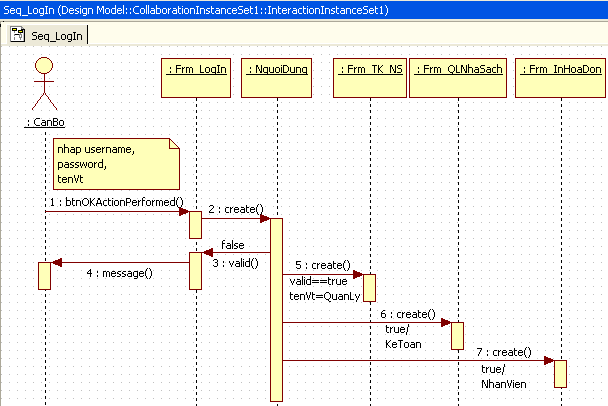
***Hướng dẫn cách tiến hành***

* *Vào* ***Model Exporer****, thêm một* ***Class diagram*** *vào* ***Design model****, đặt tên là* ***Packages***
* *Dùng chức năng  trong* ***Toolbox*** *để vẽ các packages*
* *Kéo các lớp từ* ***Model Explorer*** *vào màn hình thiết kế*
* *Chọn tất cả các phần tử trong màn hình thiết kế ( dùng chuột đánh dấu hoặc Ctrl A)*
* *Vào* ***Menu Format/Suppress Attributes*** *và* ***Suppress Operations*** *để ẩn attributes và operations giúp các lớp chỉ còn tên lớp.*
* *Dùng chức năng  trong* ***Toolbox*** *giúp vẽ các đường chỉ định gói này chứa các lớp nào.*
  1. **Sơ đồ trình tự cho các usecase**

**2.3.1- Sơ đồ trình tự cho usecase LogIn**

**Mục đích:**  Tạo giao diện và cơ chế hoạt động để cán bộ quản lý, kế toán nhân viên của nhà sach đăng nhập vào hệ thống để làm việc.

**Sơ đồ:**.



***Hướng dẫn cách tiến hành***

1. *Kích phải vào Design Model trong* ***Model Explorer***
2. *Add* ***Diagram/ Sequence Diagram****. StartUML tự động tạo một* ***CollaborationInstanceSet****, dưới đó có một* ***InteractionInstanceSet*** *và lược đồ của chúng ta nằm dưới nút này ( xem hình trên)*
3. *Đặt tên SeqLogIn*
4. *Kéo các đối tượng ( actor, class) cần có trong lược đồ từ* ***Model Explorer****.*
5. *Dùng chức năng* ***Stimulus*** *và* ***SelfStimulus*** *để vẽ các bước thao tác. StartUML không hiển thị tham số của các lời gọi hàm.*
6. *Dùng Annotaion ToolBox để đưa vào các text giải thích nếu cần.*

**Đặc tả hành vi:**

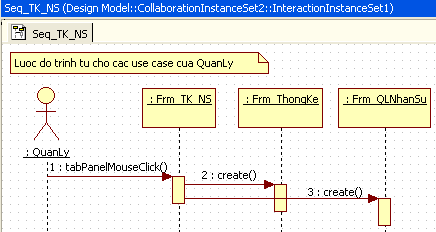
1. Người dùng gọi giao diện Login
2. Người dùng nhập dữ liệu bao gồm: username, password, vai trò
3. Hệ thống kiểm tra tính hợp lệ của người dùng
   1. Nếu dữ liệu không hợp lệ, một thông báo “Dữ liệu không hợp lệ!”
   2. Ngược lại
      1. Nếu vai trò=QuanLy, hệ thống sẽ xuất giao diện của người quản lý.
      2. Nếu vai trò=KeToan, hệ thống sẽ xuất giao diện của người kế toán.
      3. Nếu vai trò=NhanVien, hệ thống sẽ xuất giao diện của người nhân viên.

**2.3.2- Sơ đồ trình tự cho các usecase của actor QuanLy**

**Sơ đồ trình tự cho các usecase TK\_NS**

**Mục đích:** Tạo giao diện và cơ chế hoạt động tổng hợp cho người quản lý làm công việc thống kê kinh doanh và quản lý nhân sự.

**Sơ đồ:**

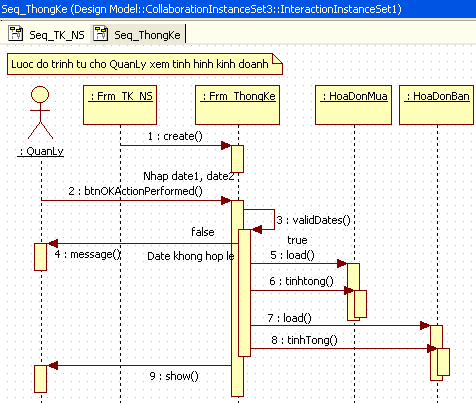


**Đặc tả hành vi:**

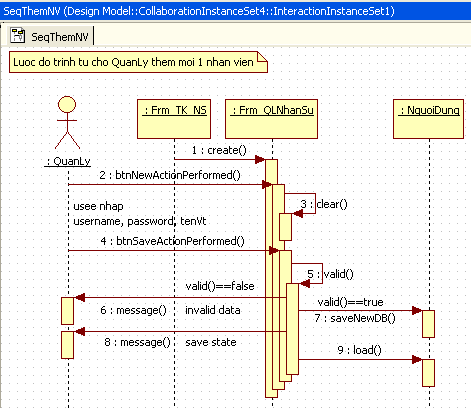
1. Người dùng chọn chức năng
2. Nếu chức năng =1 hệ thống sẽ xuất giao diện của ThongKe\_GUI

Ngược lại hệ thống sẽ xuất giao diện của QLNhanSu\_GUI

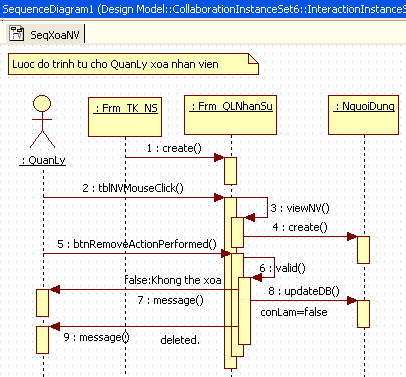
**Sơ đồ trình tự cho các usecase ThongKe**



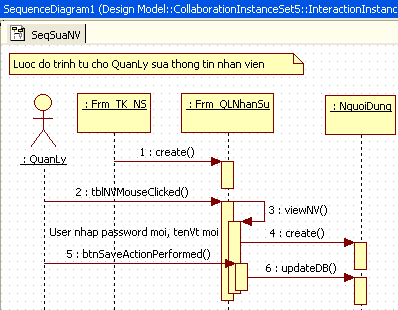
**Sơ đồ trình tự cho các usecase UC\_ThemNV**



**Lược đồ trình tự cho các usecase UC\_XoaNV**



**Lược đồ trình tự cho các usecase UC\_SuaNV**



**2.3.3- Sơ đồ trình tự cho các usecase của actor Kế toán**

Bạn tự làm

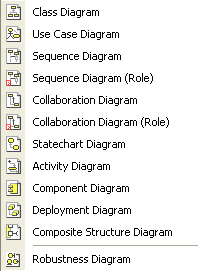
**2.3.4- Sơ đồ trình tự cho các usecase của actor NhanVien**

Bạn tự làm

**2.3.5- Sơ đồ trình tự cho các usecase của actor KhachHang**

Bạn tự làm

*Phần thiết kế này các bạn có thể làm rõ thêm thông qua các sơ đồ khác. Phần này để bạn tự xây dựng.*

**

***Sinh code***

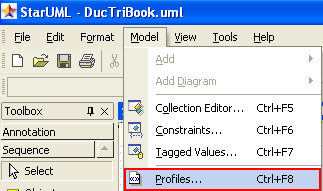
*Hầu hết các công cụ UML đề có chức năng sinh code giúp tạo khung cho các lớp đã thiết kế- tiết giảm công sức viết code- và StarUML cũng không ngoại lệ.*

*Để yêu cầu StarUML sinh code theo mõt ngôn ngữ nào đó ( thường là C++, C#, Java), người dùng phải chỉ định cho phầ mềm công cụ biết rằng mình muốn sinh code theo ngôn ngữ nào –* ***Xem lại cách ấn định Profiles*** *như đã được trình bầy ở phần đầu của tài liệu này.*

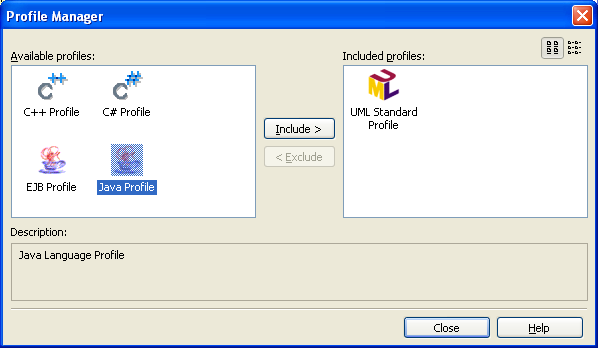
*Sau khi đã ấn định Profile cho StarUML, StarUML nhận diện các lớp cơ bản nhất của ngôn ngữ để hỗ trợ người dùng chọn kiểu của thuộc tính cũng như kiểu của trị trả về từ acc1 hành vi.*

***Cách add profile Java vào StarUML***

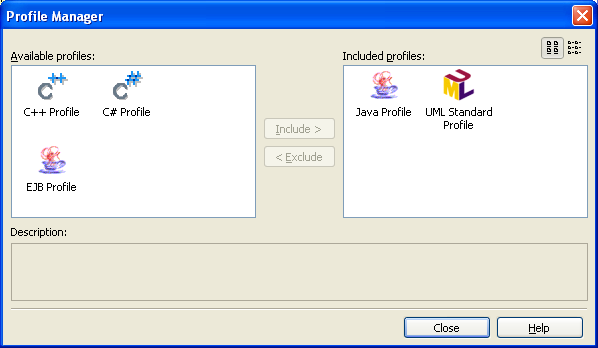
*Bước 1- Menu Model/ Profiles*

**

*Hộp thoại sau sẽ xuất hiện cho phép người dùng chọn profile được support bởi StarUML:*

**

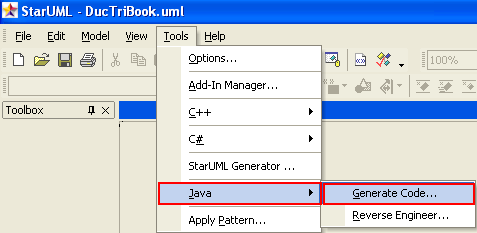
*Bước 2- Chọn Java Profile, kích nút include*

**

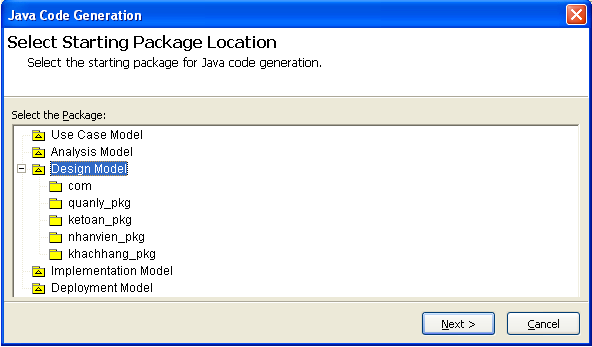
*Bước 3- Kích nút close*

***Sinh code cho mô hình***

*Bước 1- Menu*

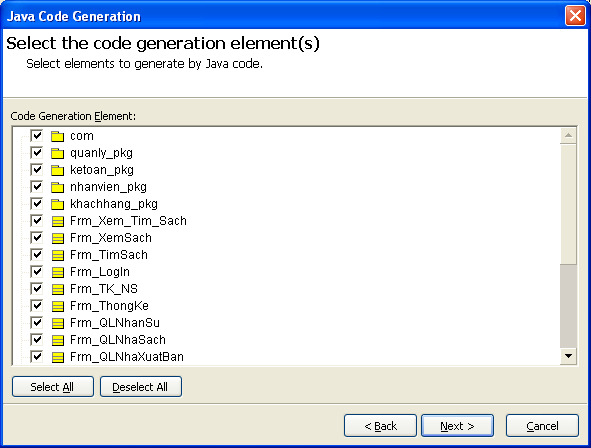
**

*Bước 2- Chọn thư mục chứa kết quả phân tích sẽ được sinh code.*

**

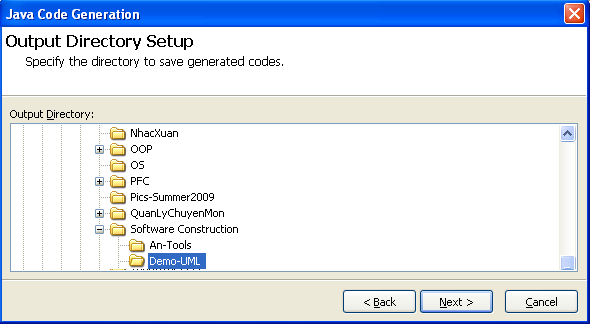
*Kích nút* ***Next***

*Bước 4- Chọn những thứ muốn sinh code (đánh dấu chọn)*



*Kích nút* ***Next***

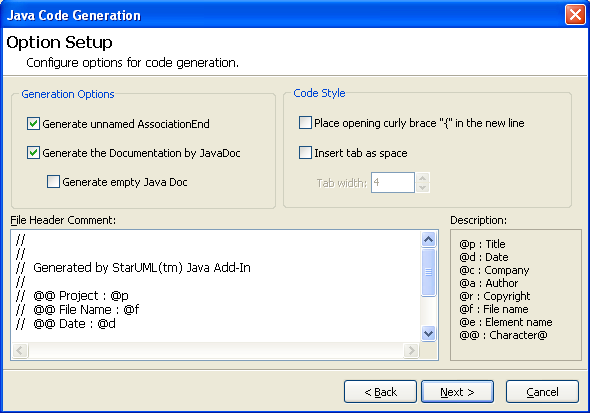
*Bước 5- Chọn thư mục chứa code*

**

*\*

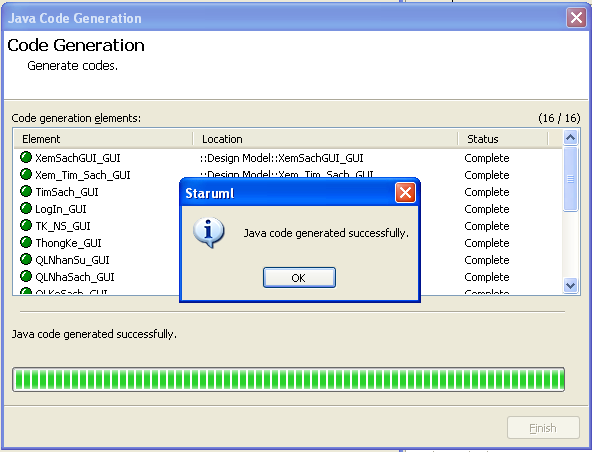
*Kích nút* ***Next***

*Bước 6- Chọn thư mục chứa code*

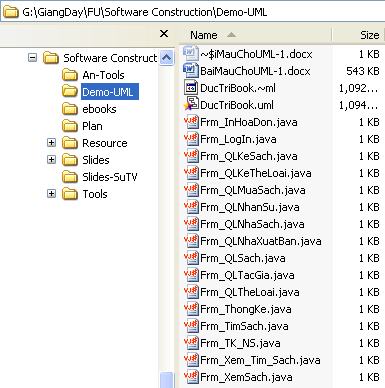
**

*Nếu đánh dấu vào mục* ***Generate unamed Association****, code của một lớp sẽ có thêm một biến tham khảo đến lớp có quan hệ.*

*Kích nút* ***Next***

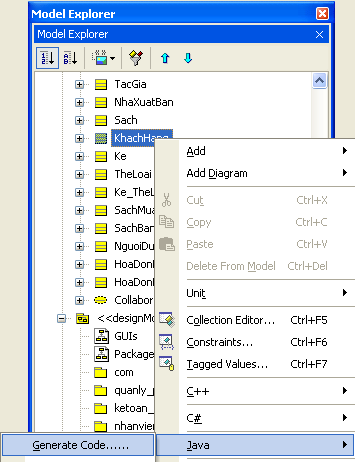
**

*Xem kết quả sinh code bằng Windows Explorer:*

**

***Chú ý: Các code sinh ra là các khai báo class với các hàm chứa code trống. Người lập trình phải tự cụ thể hoá code.***

*Bạn cũng có thể sinh code cho một đối tượng từ của số Model Explorer như sau:*

**