|  |  |  |
| --- | --- | --- |
| Macintosh HD:Users:hoangnguyen:Documents:TDC:fitlog_blue.png | TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THỦ ĐỨC  **Khoa Công Nghệ Thông Tin** | Macintosh HD:Users:hoangnguyen:Documents:TDC:logoTDC_blue.png |
|  |  |  |

**Chuyên đề phát triển web 2** | HKI – [2019 – 2020]

PRJ – REPORT

**TÌM HIỂU VỀ UML/ USE-CASE**

**Nhóm J:**

* **Thái Quang Hưng –** 17211TT3105
* **Hà Xuân Phi –** 17211TT0607
* **Trần Quốc Bữu –** 17211TT3375
* **Võ Đình Chinh –** 17211TT3587
* **Trần Bình Văn –** 17211TT0053

**BẢNG PHÂN CHIA CÔNG VIỆC**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MSSV | Họ & Tên | Công việc | Hoàn thành | Đánh giá |
| 17211TT0053 | Trần Bình Văn | * Demo UML/ Use-Case * Keywords * Tổng hợp, chỉnh sửa, bổ sung chi tiết báo cáo | 100% | Trình bày nhiều ảnh minh họa, đầy đủ |
| 17211TT0607 | Hà Xuân Phi | * Tìm hiểu sơ về biểu đồ hoạt động, biểu đồ tương tác, biểu đồ trạng thái | 100% | Trình bày đầy đủ, chi tiết, nhiều hình ảnh minh họa |
| 17211TT3375 | Trần Quốc Bữu | * Biểu đồ Use – Case (chi tiết) | 100% | Trình bày đầy đủ, nhiều hình ảnh minh họa |
| 17211TT3587 | Võ Đình Chinh | * Giới thiệu về UML * Mục đích UML * Lịch sử phát triển UML * Công cụ UML | 100% | Trình bày đầy đủ, nhiều hình ảnh |
| 17211TT3105 | Thái Quang Hưng | * Tìm hiểu sơ về biểu đồ cấu trúc tĩnh, biểu đồ thực thi | 100% | Trình bày đầy đủ, nhiều hình ảnh |

**MỤC LỤC**

[**KEYWORDS** 1](#_Toc24197378)

[**I.** **Giới thiệu về UML** 2](#_Toc24197379)

[**II.** **Mục Đích** 2](#_Toc24197380)

[**III.** **Lịch sử phát triển của UML** 3](#_Toc24197381)

[**IV.** **Các công cụ UML** 4](#_Toc24197382)

[**V.** **Một số dạng biểu đồ UML** 5](#_Toc24197383)

[**1.** **Biểu đồ Use Case (Use case diagram)** 5](#_Toc24197384)

[**2.** **Biểu đồ hoạt động (Activity Diagram)** 8](#_Toc24197385)

[**3.** **Biểu đồ tương tác (Interaction Diagram)** 9](#_Toc24197386)

[**4.** **Biểu đồ trạng thái (Statechart Diagram)** 11](#_Toc24197387)

[**5.** **Biểu đồ cấu trúc tĩnh (Static Structure Diagram)** 11](#_Toc24197388)

[**6.** **Biểu đồ thực thi (Implementation Diagrams)** 13](#_Toc24197389)

[**VI.** **Demo UML/Use-Case ra code** 14](#_Toc24197390)

[**TÀI LIỆU THAM KHẢO** 21](#_Toc24197391)

**DANH MỤC HÌNH ẢNH**

[Hình 1. Logo UML 2](#_Toc24197337)

[Hình 2. UML là ngôn ngữ hợp nhất các mô hình khác nhau 3](#_Toc24197338)

[Hình 3. Lịch sử phát triển UML 3](#_Toc24197339)

[Hình 4. Các công cụ mã nguồn mở 4](#_Toc24197340)

[Hình 5. Các công cụ thương mại 4](#_Toc24197341)

[Hình 6. Ví dụ cơ bản về sơ đồ Use Case. 5](#_Toc24197342)

[Hình 7. Kí hiệu của Actor. 5](#_Toc24197343)

[Hình 8. Một dạng khác của Actor. 5](#_Toc24197344)

[Hình 9. Kí hiệu của Use Case . 6](#_Toc24197345)

[Hình 10. Ví dụ quan hệ "Include" trong Use Case Diagram. 6](#_Toc24197346)

[Hình 11. Ví dụ về quan hệ "Extend" trong Use Case Diagram. 7](#_Toc24197347)

[Hình 12. Ví dụ quan hệ kế thừa trong Use Case Diagram. 7](#_Toc24197348)

[Hình 13.Ví dụ bản vẽ use-case ATM 7](#_Toc24197349)

[Hình 14. Ví dụ cơ bản về Activity Diagram. 8](#_Toc24197350)

[Hình 15. Kí hiệu nút Start. 8](#_Toc24197351)

[Hình 16. Kí hiệu nút End. 9](#_Toc24197352)

[Hình 17. Ví dụ về Activity. 9](#_Toc24197353)

[Hình 18. Kí hiệu Branch. 9](#_Toc24197354)

[Hình 19. Kí hiệu về Fork. 9](#_Toc24197355)

[Hình 20. Kí hiệu về Join. 9](#_Toc24197356)

[Hình 21. Ví dụ về dạng biểu đồ Interaction Diagram. 10](#_Toc24197357)

[Hình 22. Kí hiệu Objets trong Interaction Diagram. 10](#_Toc24197358)

[Hình 23. Kí hiệu về Stimulus trong Interaction Diagram. 10](#_Toc24197359)

[Hình 24. Các kí hiệu UML thể hiện bắt đầu, kết thúc, sự kiện và trạng thái của đối tượng 11](#_Toc24197360)

[Hình 25. Ví dụ về biểu đồ trạng thái. 11](#_Toc24197361)

[Hình 26. Ví dụ về Class Diagram. 12](#_Toc24197362)

[Hình 27. Ví dụ về Object Diagram. 12](#_Toc24197363)

[Hình 28. Ví dụ về biểu đồ thành phần. 13](#_Toc24197364)

[Hình 29. Ví dụ về Deployment Diagram. 13](#_Toc24197365)

[Hình 30. Demo bước 1 14](#_Toc24197366)

[Hình 31. Demo bước 2 15](#_Toc24197367)

[Hình 32. Demo bước 3 15](#_Toc24197368)

[Hình 33. Demo bước 4 16](#_Toc24197369)

[Hình 34. Demo bước 5 16](#_Toc24197370)

[Hình 35. Demo bước 6 17](#_Toc24197371)

[Hình 36. Demo bước 7 18](#_Toc24197372)

[Hình 37. Demo bước 8 18](#_Toc24197373)

[Hình 38. Kết quả từ demo 1 19](#_Toc24197374)

[Hình 39. Kết quả từ demo 2 19](#_Toc24197375)

[Hình 40. Kết quả từ demo 3 20](#_Toc24197376)

[Hình 41. Kết quả từ demo 4 20](#_Toc24197377)

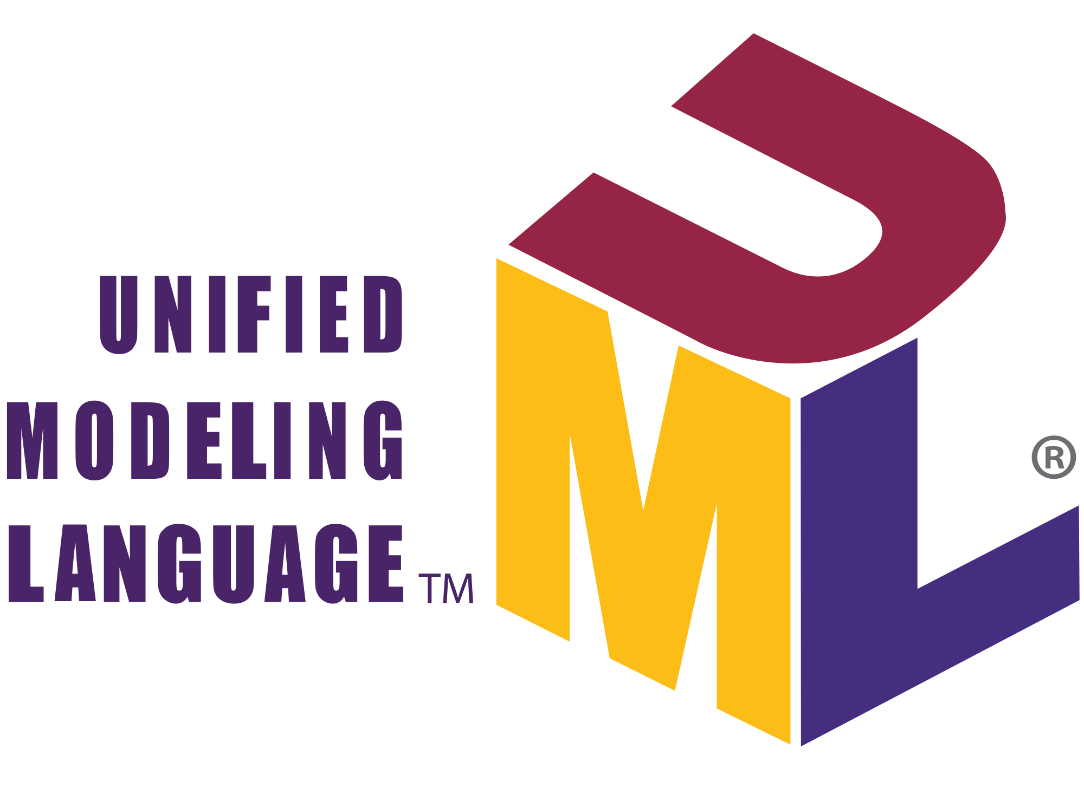
# **KEYWORDS**

* Unified Modeling Language (Ngôn ngữ mô hình hóa thống nhất)
* Object-Oriented (Hướng đối tượng)
* Use case diagram (Biểu đồ use case)
* Include (Quan hệ bao gồm trong biểu đồ use case)
* Extend (Quan hệ mở rộng trong biểu đồ use case)
* Generalization (Quan hệ tổng quát trong biểu đồ use case)
* Activity Diagram (Biểu đồ hoạt động)
* Interaction Diagram (Biểu đồ tương tác)
* Statechart Diagram (Biểu đồ trạng thái)
* Subsystem (Hệ thống con)
* Static Structure Diagram (Biểu đồ cấu trúc tĩnh)
* Class Diagram (Biểu đồ lớp)
* Associated (Kết nối với nhau)
* Dependent (Phụ thuộc)
* Specialized (Chuyên biệt hóa)
* Packaged (Đóng gói)
* Object Diagram (Biểu đồ đối tượng)
* Implementation Diagram (Biểu đồ thực thi)
* Component Diagram (Biểu đồ thành phần)
* Deployment Diagram (Biểu đồ triển khai)

# **Giới thiệu về UML**

UML (Unified Modeling Language) ngôn ngữ mô hình hóa thống nhất là ngôn ngữ dành cho việc đặc tả, hình dung, xây dựng và làm tài liệu của các hệ thống phần mềm.

Đây là ngôn ngữ thay thế xứng đáng cho Booch, OOSE/Jacobson, OMT,...

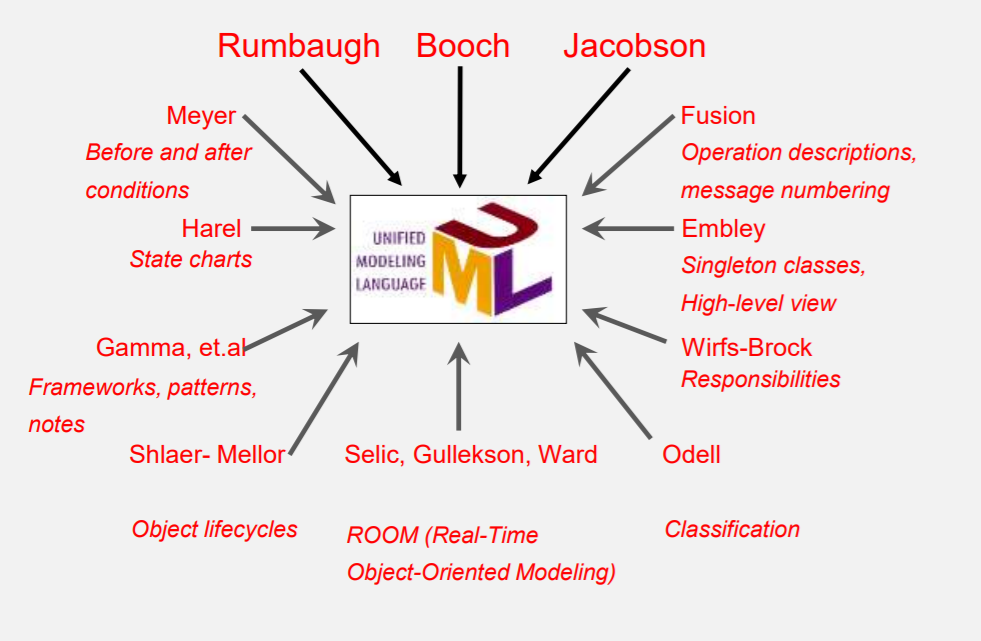


Hình 1. Logo UML

# **Mục Đích**

UML cung cấp cho người dùng một ngôn ngữ mô hình hoá trực quan sẵn sàng:

* UML là ngôn ngữ trực quan giúp công việc phát triển được xử lý đồng nhất, ít lỗi
* Phù hợp thực tế
* Khả năng mở rộng, chuyên môn hóa để khám phá giá trị cốt lõi
* Có thể kết nối trực tiếp với rất nhiều ngôn ngữ lập trình như C++, Java, Visual Basic…
* Áp dụng được cho tất cả các pha của quy trình phát triển phần mềm.



Hình 2. UML là ngôn ngữ hợp nhất các mô hình khác nhau

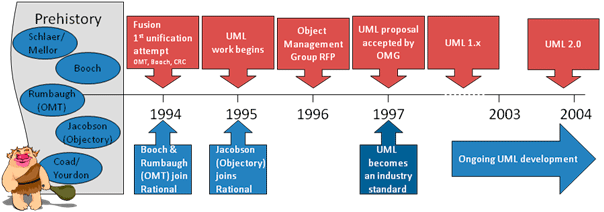
# **Lịch sử phát triển của UML**

Số lượng các phương pháp luận hướng đối tượng gia tăng từ dưới 10 đến 50 trong khoảng những năm 1989 đến 1994, quá nhiều sự lựa chọn, không thống nhất

Vào 10/1994, Rumbaugh hợp tác với công ty Booch (Rational Sofware Corporation) để kết hợp phương pháp Booch và phương pháp OMT. Tạo ra một bản phác thảo về phương pháp có tên là Unified Process vào 10/1995.

Cũng trong năm 1995, Jacobson đã tích hợp phương pháp này với OOSE. Những tài liệu đầu tiên về UML đã được trình làng vào trong năm 1996.

Phiên bản 1.0 của UML đã được công bố vào tháng giêng 1997



Hình 3. Lịch sử phát triển UML

# **Các công cụ UML**

Công cụ mã nguồn mở:

* EclipseUML
* UmlDesigner
* StarUML
* Argo UML



Hình . Các công cụ mã nguồn mở

Công cụ thương mại:

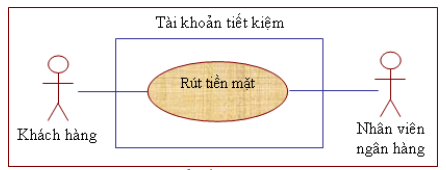
* Enterprise Architect
* IBM Rational Software Architect
* Microsoft Visio
* Visual Paradigm for UML
* SmartDraw...



Hình 5. Các công cụ thương mại

1. **Một số dạng biểu đồ UML**
2. **Biểu đồ Use Case (Use case diagram)**

* Use Case được mô tả trong ngôn ngữ UML qua biểu đồ Use Case (Use Case Diagram). Một biểu đồ Use Case chứa các phần tử mô hình biểu thị hệ thống, tác nhân cũng như Use Case.
* Một biểu đồ Use Case sẽ thể hiện rõ những tiêu chí sau:
  + Hệ thống
  + Tác nhân
  + Use Case



Hình . Ví dụ cơ bản về sơ đồ Use Case.

* **Hệ thống:** biểu diễn ranh giới giữa bên trong và bên ngoài của một chủ thể trong phần mềm.Một hệ thống ở trong biểu đồ use case không nhất thiết là một hệ phần mềm; nó có thể là một chiếc máy, …
* **Tác nhân (Actor):** là người dùng của hệ thống, một tác nhân có thể là một người dùng thực hoặc các hệ thống máy tính khác có công việc.



Hình 7. Kí hiệu của Actor.



Hình 8. Một dạng khác của Actor.

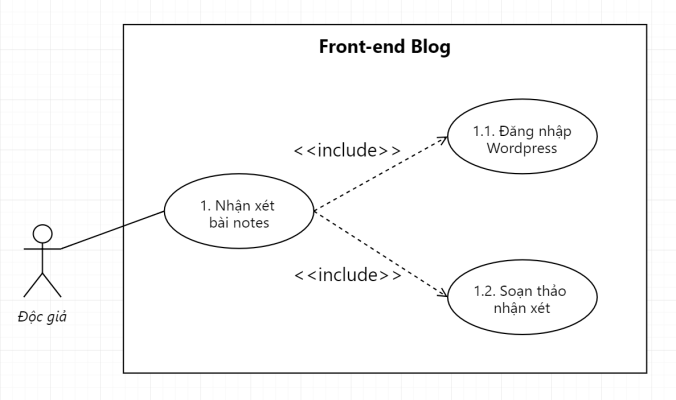
* **Các Use Case:** Đây là thành phần cơ bản. Các use case được biểu thị bởi các hình elip.Tên các use case thể hiện một chức năng xác định của hệ thống.



Hình 9. Kí hiệu của Use Case .

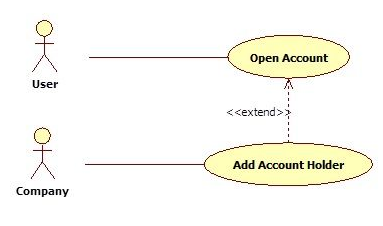
* Các mối quan hệ trong sơ đồ Use Case:

1. **Include (QH bao gồm):** là quan hệ giữa các Use Case với nhau, nó mô tả việc một Use Case lớn được chia ra thành các Use Case nhỏ để dễ cài đặt hoặc thể hiện sự dùng lại.
   * Khi thực hiện một chức năng sẽ bắt buộc thực hiện chức năng đi theo.
   * Ký hiệu: <<include>>



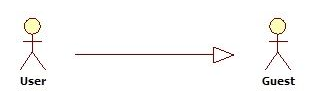
Hình 10. Ví dụ quan hệ "Include" trong Use Case Diagram.

1. **Extend (QH Mở rộng)** Mô tả quan hệ giữa 2 Use Case. Được sử dụng khi có một Use Case được tạo ra để bổ sung chức năng cho một Use Case khác và được sử dụng trong một điều kiện nhất định.
   * Khi thực hiện một chức năng sẽ có thể, tùy chọn thực hiện chức năng đi theo.
   * Ký hiệu: <<extend>>

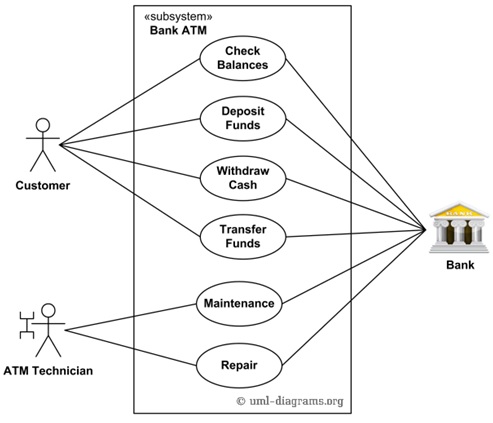


Hình 11. Ví dụ về quan hệ "Extend" trong Use Case Diagram.

1. **Generalization (QH Tổng quát):** được sử dụng để thể hiện quan hệ thừa kế giữa các Actor hoặc giữa các Use Case với nhau.



Hình 12. Ví dụ quan hệ kế thừa trong Use Case Diagram.



Hình .Ví dụ bản vẽ use-case ATM

Các bước để xây dựng một bản vẽ Use Case:

**+ Bước 1: Tìm các Actor**

Trả lời các câu hỏi sau để xác định Actor cho hệ thống:

Như vậy có 03 Actor: Customer, ATM Technician và Bank

**+ Bước 2: Tìm các Use-Case**

Trả lời câu hỏi các Actor sử dụng chức năng gì trong hệ thống? Xác định được use-case cho hệ thống

**+ Bước 3: Xác định các quan hệ**

Phân tích và các định các quan loại hệ giữa các Actor và Use Case, giữa các Actor với nhau, giữa các Use Case với nhau sau đó nối chúng lại sẽ được bản vẽ Use Case.

**\*\*Đặc tả Use Case**

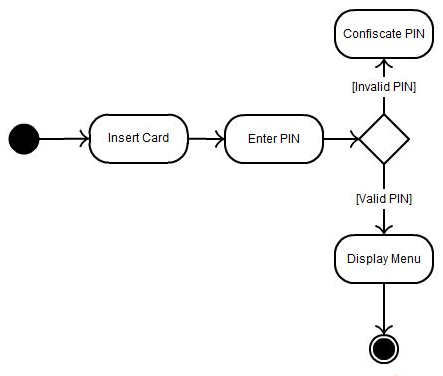
Nhìn vào bản vẽ trên nhận biết hệ thống cần những chức năng gì và ai sử dụng. Nhưng chưa biết được nó vận hành ra sao? Sử dụng chúng như thế nào? Phải đặc tả để có thể hiểu rõ hơn.

**Cách 1: Viết đặc tả cho các Use Case**

**Cách 2: Sử dụng các bản vẽ để đặc tả**

1. **Biểu đồ hoạt động (Activity Diagram)**

* Activity Diagram là bản vẽ tập trung vào mô tả các hoạt động, luồng xử lý bên trong hệ thống.



Hình 14. Ví dụ cơ bản về Activity Diagram.

* Các kí hiệu của Activity Diagram:
  + Nút Start & End:
    - Nút Start thể hiện điểm bắt đầu quá trình.



Hình 15. Kí hiệu nút Start.

* + - Nút End thể hiện điểm kết thúc quá trình.



Hình 16. Kí hiệu nút End.

* + Activity: mô tả một hoạt động trong hệ thống. Các hoạt động này do các đối tượng thực hiện.



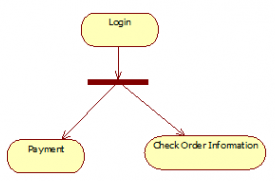
Hình 17. Ví dụ về Activity.

* + Branch: thể hiện rẽ nhánh trong mệnh đề điều kiện.



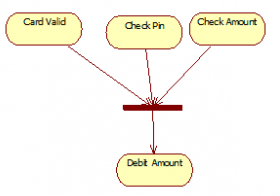
Hình 18. Kí hiệu Branch.

* + Fork: thể hiện cho trường hợp thực hiện xong một hoạt động rồi rẽ nhánh nhiều hoạt động khác



Hình 19. Kí hiệu về Fork.

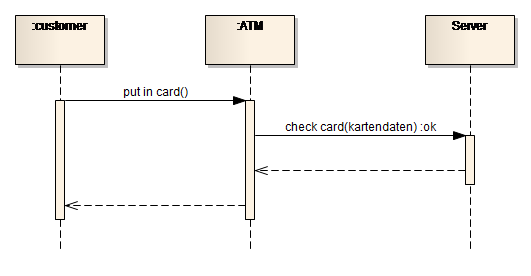
* + Join: Cùng ký hiệu với Fork nhưng thể hiện trường hợp phải thực hiện hai hay nhiều hành động trước rồi mới thực hiện hành động tiếp theo.



Hình 20. Kí hiệu về Join.

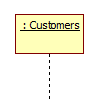
1. **Biểu đồ tương tác (Interaction Diagram)**

* Bản vẽ mô tả sự tương tác của các đối tượng để tạo nên các chức năng của hệ thống.
* Phù hợp với việc sử dụng để thiết kế và cài đặt chức năng cho hệ thống phần mềm.
* Trong biểu đồ tương tác (Interaction Diagrams) thì chia thành 2 dạng biểu đồ nhỏ là :
  + Biểu đồ trình tự (Sequence Diagram).
  + Biểu đồ giao tiếp/cộng tác (Communication/Collaboration Diagram).



Hình 21. Ví dụ về dạng biểu đồ Interaction Diagram.

* Các thành phần của Sequence Diagram:
  + Objects: mô tả một đối tượng trong hệ thống. Để phân biệt với Class, Object có dấu “:” phía trước tên của nó.
  + Đường gạch chấm bên dưới đối tượng thể hiện thời gian tồn tại của đối tượng.



Hình 22. Kí hiệu Objets trong Interaction Diagram.

* + Stimulus (message): Stimulus thể hiện thông điệp từ một đối tượng này tương tác với một đối tượng khác.



Hình 23. Kí hiệu về Stimulus trong Interaction Diagram.

* + Axes: Trục tọa độ, trục ngang thể hiện các đối tượng, trục đứng thể hiện thời gian.

1. **Biểu đồ trạng thái (Statechart Diagram)**

* Biểu đồ trạng thái nắm bắt vòng đời của các đối tượng, các hệ thống con và các hệ thống.
* Biểu đồ trạng thái thể hiện những khía cạnh khi xem xét trạng thái của một đối tượng

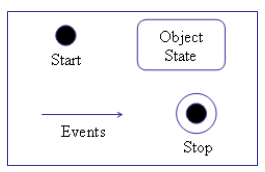
- Trạng thái ban đầu

- Một số trạng thái ở giữa

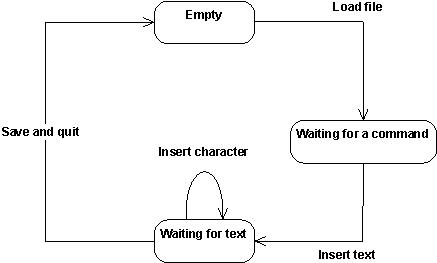
- Một hoặc nhiều trạng thái kết thúc

- Sự biến đổi giữa các trạng thái

- Những sự kiện biến đổi từ một trạng thái này sang trạng thái khác



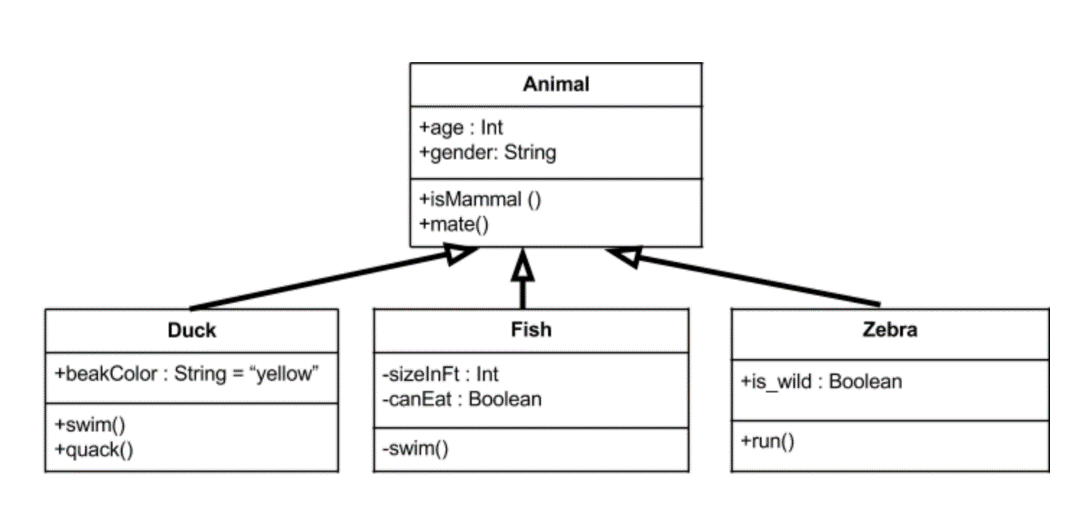
Hình . Các kí hiệu UML thể hiện bắt đầu, kết thúc, sự kiện và trạng thái của đối tượng



Hình . Ví dụ về biểu đồ trạng thái.

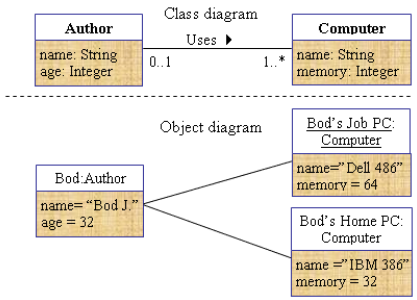
1. **Biểu đồ cấu trúc tĩnh (Static Structure Diagram)**

* Biểu đồ cấu trúc tĩnh (Static Structure Diagrams) chia thành 2 dạng biểu đồ phổ biến là:
  + Biểu đồ lớp (Class Diagram)
  + Biểu đồ đối tượng (Object Diagram)
* Biểu đồ lớp (Class Diagram): chỉ ra cấu trúc tĩnh của các lớp trong hệ thống. Các lớp có thể quan hệ với nhau trong nhiều dạng thức:
  + Liên kết (associated - được nối kết với nhau).
  + Phụ thuộc (dependent - một lớp này phụ thuộc vào lớp khác).
  + Chuyên biệt hóa (specialized - một lớp này là một kết quả chuyên biệt hóa của lớp khác).
  + Đóng gói (packaged – hợp với nhau thành một đơn vị)



Hình 26. Ví dụ về Class Diagram.

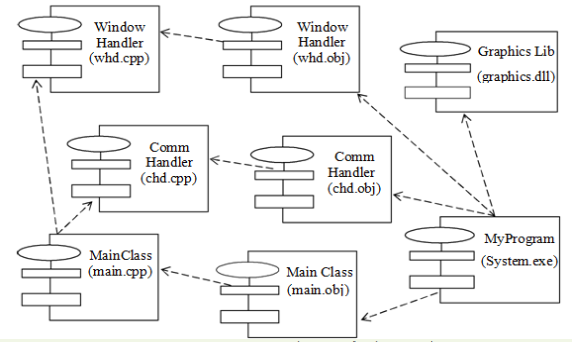
* Biểu đồ đối tượng (Object Diagram): phiên bản của biểu đồ lớp và thường cũng sử dụng các ký hiệu như biểu đồ lớp.
* Hai biểu đồ này khác nhau ở chỗ biểu đồ đối tượng chỉ ra một loạt các đối tượng thực thể của lớp, thay vì các lớp.



Hình 27. Ví dụ về Object Diagram.

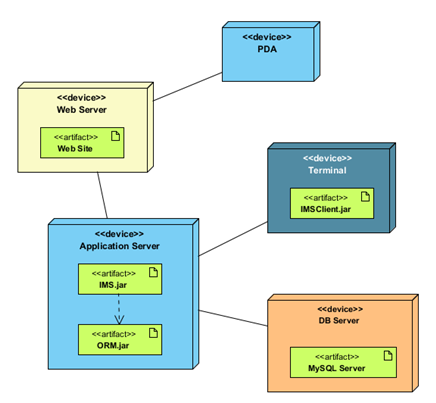
1. **Biểu đồ thực thi (Implementation Diagrams)**

* Biểu đồ thực thi chia thành 2 dạng biểu đồ nhỏ là:
  + Biểu đồ thành phần (Component Diagram).
  + Biểu đồ triển khai (Deployment Diagram).
* Biểu đồ thành phần (Component Diagram): là biểu đồ mô tả các thành phần và sự phụ thuộc của chúng trong hệ thống.



Hình 28. Ví dụ về biểu đồ thành phần.

* Biểu đồ triển khai (Deployment Diagram): thể hiện rõ kiến trúc triển khai nên nó sẽ ảnh hưởng đến sự thiết kế, phát triển,… của hệ thống

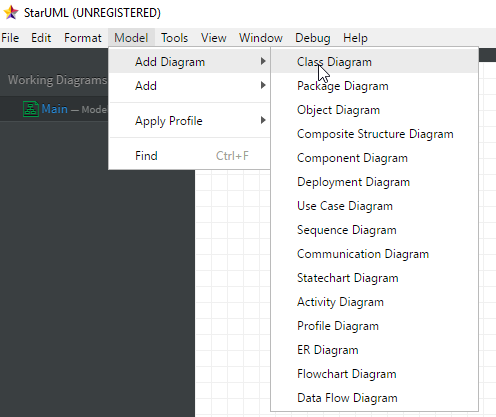


Hình 29. Ví dụ về Deployment Diagram.

1. **Demo UML/Use-Case ra code**

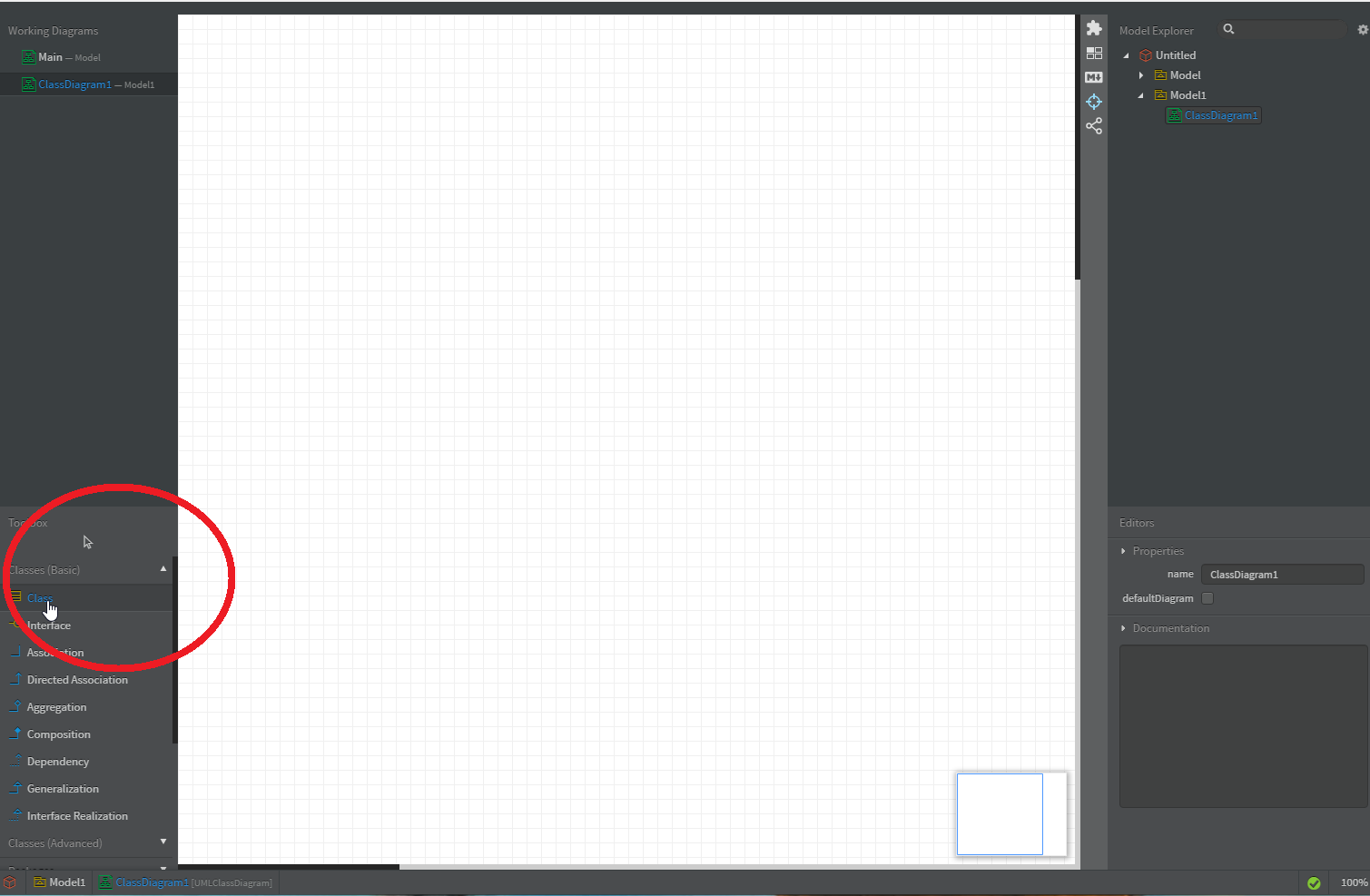
Để có thể dễ hiểu hơn, ta sẽ thực hiện vẽ một sơ đồ Class (Class Diagram) sau đó tạo ra code C++ từ các Class vẽ nên

B1: Ta mở chương trình StarUML hoặc bất kì chương trình vẽ UML nào ra, chọn Model -> Add Diagram -> Class Diagram

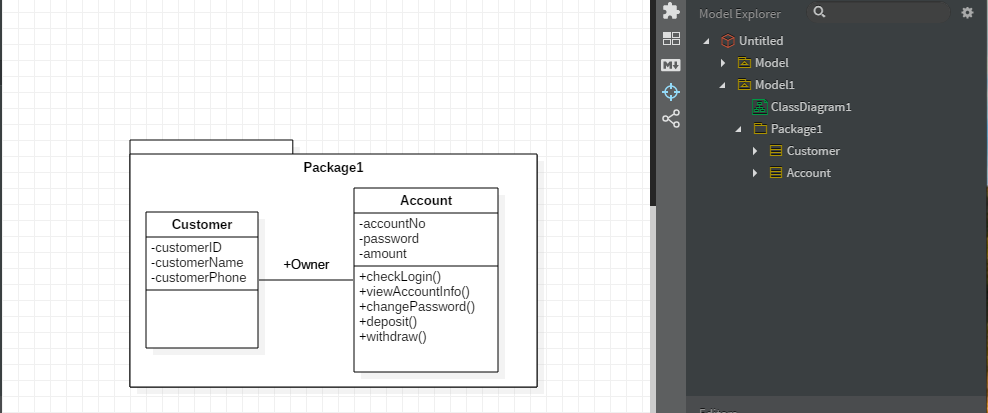


Hình . Demo bước 1

Bước 2: Ta bắt đầu vẽ sơ đồ Class bằng các ký hiệu bên tay trái phía dưới

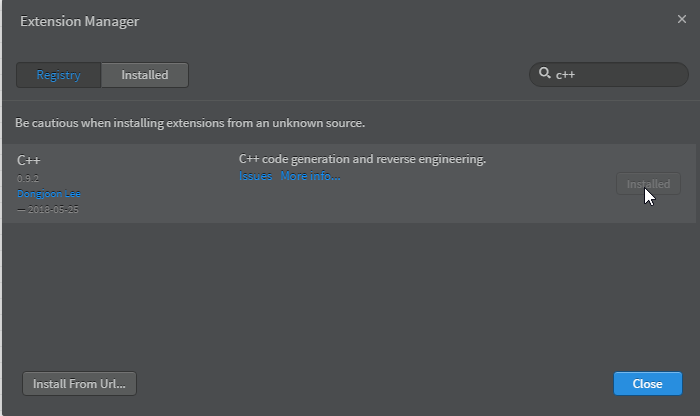


Hình . Demo bước 2



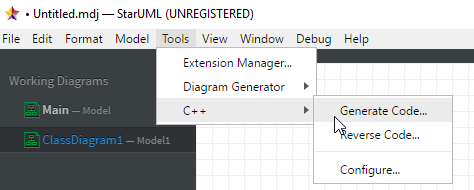
Hình . Demo bước 3

Bước 4: Sau khi vẽ xong ta chọn Tools 🡪 Extension Manager… 🡪 rồi chọn ngôn ngữ ta muốn xuất code ra (ở đây ta ví dụ code C++ nên ta tìm kiếm rồi chọn Install)



Hình . Demo bước 4

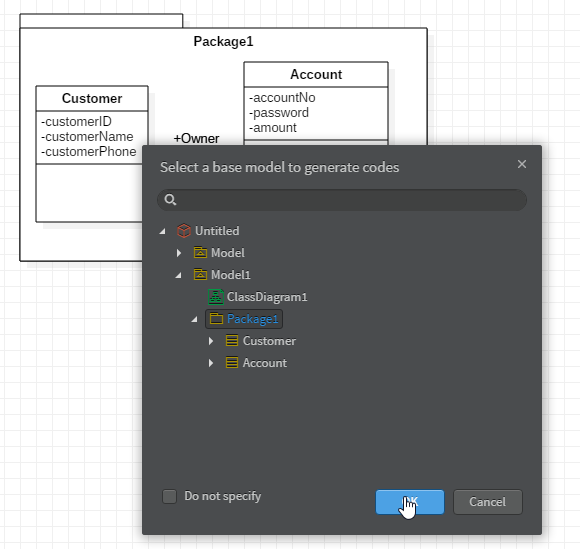
Bước 5: Sau khi ta cài đặt xong Extension C++ ta chọn Reload, khởi động lại chương trình và chọn như sau để tạo code



Hình . Demo bước 5

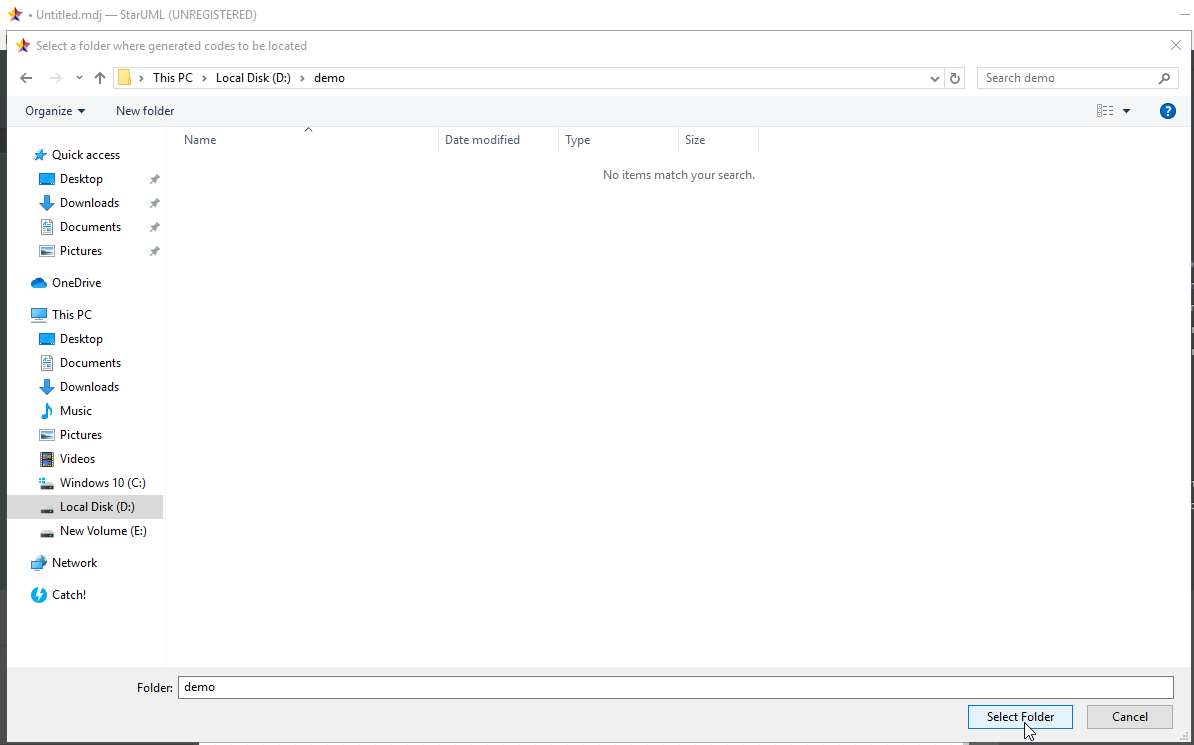
Bước 6: Ta chọn Package mà ta để các class ở trong, nhấn OK

\*\*Lưu ý ở bước này các Class ta vẽ phải được gom lại và đặt trong 1 package mới có thể tạo ra code



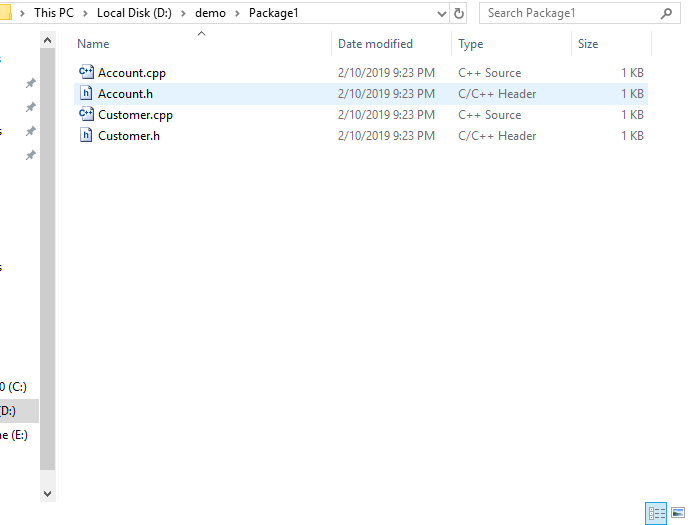
Hình . Demo bước 6

Bước 7: Ta chọn đường dẫn lưu code

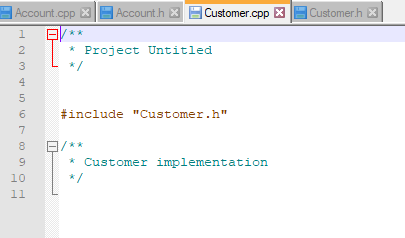


Hình . Demo bước 7

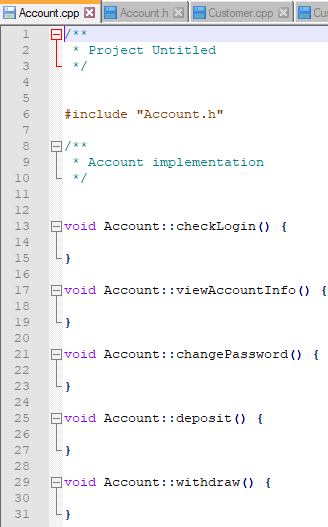
Bước 8: Kiểm tra kết quả



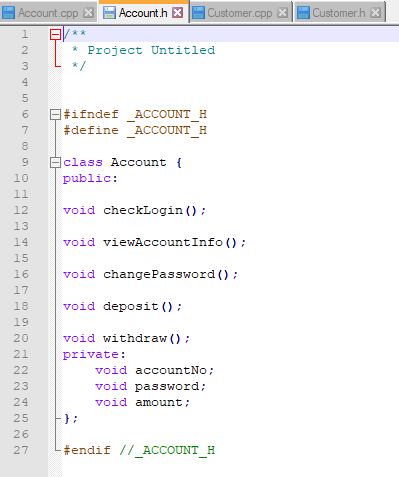
Hình . Demo bước 8



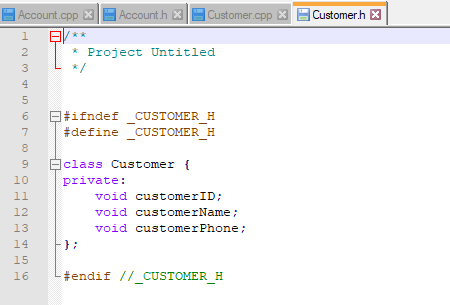
Hình . Kết quả từ demo 1



Hình . Kết quả từ demo 2



Hình . Kết quả từ demo 3



Hình . Kết quả từ demo 4

# **TÀI LIỆU THAM KHẢO**

<https://www.youtube.com/watch?v=HHCBM9cGXbU>

…và các tài liệu liên quan