## 1. Object Oriented Paradigm

**1.1 Phân tích hướng đối tượng**

Phân tích hướng đối tượng (OOA) là thủ tục xác định yêu cầu về kỹ thuật phần mềm và phát triển các đặc tả phần mềm theo mô hình đối tượng của hệ thống phần mềm, bao gồm các đối tượng tương tác.

Sự khác biệt chính giữa phân tích hướng đối tượng và các dạng phân tích khác là trong cách tiếp cận hướng đối tượng, các yêu cầu được tổ chức xung quanh các đối tượng, các đối tượng này tích hợp cả dữ liệu và các chức năng. Chúng được mô phỏng theo các đối tượng thế giới thực mà hệ thống tương tác. Trong phương pháp phân tích truyền thống, hai khía cạnh - chức năng và dữ liệu - được xem xét riêng.

Các nhiệm vụ chính trong phân tích hướng đối tượng (OOA) là:

* Xác định đối tượng
* Tổ chức các đối tượng bằng cách tạo sơ đồ mô hình đối tượng
* Xác định internal của các đối tượng, hoặc đối tượng thuộc tính
* Xác định hành vi của các đối tượng, tức là hành động đối tượng
* Mô tả cách các đối tượng tương tác

Các mô hình phổ biến được sử dụng trong OOA là use case và object model.

**1.2 Thiết kế hướng đối tượng**

Thiết kế hướng đối tượng (OOD) bao gồm việc thực hiện mô hình khái niệm được tạo ra trong quá trình phân tích hướng đối tượng. Trong OOD, các khái niệm trong mô hình phân tích, độc lập về công nghệ, được ánh xạ vào việc cài đặt các lớp, các ràng buộc được xác định và các giao diện được thiết kế, dẫn đến một mô hình cho miền giải pháp, tức là một mô tả chi tiết về cách thức của hệ thống được xây dựng trên công nghệ cụ thể.

Chi tiết thực hiện bao gồm:

* Cơ cấu lại dữ liệu lớp (nếu cần),
* Thực thi các phương thức, tức là các cấu trúc dữ liệu nội bộ và các thuật toán,
* Thực hiện control
* Thực hiện các liên kiết.

**1.3 Lập trình hướng đối tượng**

Lập trình hướng đối tượng (OOP) là một mô hình lập trình dựa trên các đối tượng (có cả dữ liệu và phương thức) nhằm kết hợp các ưu điểm của việc modul hóa và khả năng tái sử dụng. Đối tượng thường là instance của các lớp, được sử dụng để tương tác với nhau để thiết kế các ứng dụng và các chương trình máy tính.

Các tính năng quan trọng của lập trình hướng đối tượng là:

* Cách tiếp cận từ dưới lên trong thiết kế chương trình
* Các chương trình được tổ chức xung quanh các đối tượng, được nhóm lại trong các lớp.
* Tập trung vào dữ liệu với các phương thức thực thi dựa trên dữ liệu của đối tượng
* Tương tác giữa các đối tượng thông qua chức năng
* Khả năng tái sử dụng thiết kế thông qua việc tạo ra các lớp mới bằng cách thêm các tính năng vào các lớp hiện có

Một số ví dụ về các ngôn ngữ lập trình hướng đối tượng là C ++, Java, Smalltalk, Delphi, C #, Perl, Python, Ruby và PHP.

## 2. OOAD - Object Model

**2.1 Đối tượng và class**

Các khái niệm về các đối tượng và các lớp học gắn liền với nhau và tạo thành nền tảng của mô hình hướng đối tượng.

**2.1.1 Object**

Một đối tượng là một phần tử trong thế giới thực trong một môi trường hướng đối tượng có thể có một sự tồn tại về mặt vật lý hay khái niệm. Mỗi đối tượng có:

* Đặc trưng phân biệt nó với các đối tượng khác trong hệ thống.
* Trạng thái xác định các thuộc tính đặc trưng của một đối tượng cũng như các giá trị của các thuộc tính mà đối tượng giữ.
* Hành vi đại diện cho các hoạt động có thể nhìn thấy bên ngoài được thực hiện bởi một đối tượng trong các thay đổi trong trạng thái của nó.

Đối tượng có thể được mô phỏng theo yêu cầu của ứng dụng. Một vật thể có thể có một sự tồn tại vật lý, như một khách hàng, một chiếc xe, vv; Hoặc sự tồn tại khái niệm phi vật thể, như dự án, quy trình, v.v

**2.1.2 Lớp**

Một lớp đại diện cho một bộ sưu tập của các đối tượng có cùng đặc tính đặc trưng có hành vi chung. Nó đưa ra kế hoạch chi tiết hoặc mô tả các đối tượng có thể được tạo ra từ nó. Việc tạo ra một đối tượng như một thành viên của một lớp được gọi là instantiation. Do đó, đối tượng là một thể hiện của một lớp.

Các thành phần của một lớp là:

* Một bộ các thuộc tính cho các đối tượng sẽ được thể hiện bởi lớp. Nói chung, các đối tượng khác nhau của một lớp có một số khác biệt trong các giá trị các thuộc tính. Thuộc tính thường được gọi là dữ liệu lớp.
* Một tập hợp các hoạt động miêu tả hành vi của các đối tượng của lớp. Các hoạt động cũng được gọi là hàm hoặc phương thức.

Thí dụ

Chúng ta hãy xem xét một lớp đơn giản, Circle, đại diện cho hình tròn trong không gian hai chiều. Các thuộc tính của lớp này có thể được xác định như sau:

* X-coord, để biểu thị x-tọa độ của trung tâm
* Y-coord, để biểu thị tọa độ y của tâm
* a để biểu thị bán kính của vòng tròn

Một số hoạt động của nó có thể được định nghĩa như sau:

* findArea(), phương thức tính diện tích
* findCircumference(), phương thức để tính chu vi
* scale(), phương thức tăng hoặc giảm bán kính

Trong quá trình khởi tạo, các giá trị được gán cho ít nhất một vài thuộc tính. Nếu chúng ta tạo ra một đối tượng my\_circle, chúng ta có thể gán các giá trị như x-coord: 2, y-coord: 3, và a: 4 để mô tả trạng thái của nó. Bây giờ, nếu phương thức scale() được thực hiện trên my\_circle với một hệ số là 2, giá trị của biến a sẽ trở thành 8. Thao tác này mang lại sự thay đổi trạng thái của my\_circle, tức là, đối tượng đã thể hiện được một hành vi nhất định.

**2.2 Đóng gói và Ẩn dữ liệu**

**2.2.1 Đóng gói**

Đóng gói là quá trình ràng buộc cả thuộc tính và phương thức với nhau trong một lớp. Thông qua việc đóng gói, các chi tiết nội bộ của một lớp học có thể được ẩn từ bên ngoài. Nó cho phép các thành phần của lớp được truy cập từ bên ngoài chỉ thông qua giao diện được cung cấp bởi lớp.

**2.2.2 Ẩn dữ liệu**

Thông thường, một lớp được thiết kế sao cho dữ liệu (thuộc tính) của nó chỉ có thể được truy cập theo các phương thức của nó và được cách ly khỏi các truy cập trực tiếp từ bên ngoài. Quá trình được gọi là ẩn dữ liệu hoặc ẩn thông tin.

Thí dụ

Trong lớp Circle, ẩn dữ liệu có thể được kết hợp bằng cách làm cho các thuộc tính không nhìn thấy được từ bên ngoài lớp học và thêm hai phương thức nữa vào lớp để truy cập dữ liệu lớp, cụ thể là:

setValues​​(), phương pháp để gán các giá trị cho x-coord, y-coord, và a

getValues(), phương thức lấy giá trị của x-coord, y-coord, và a

Ở đây, dữ liệu cá nhân của đối tượng my\_circle không thể được truy cập trực tiếp bằng bất kỳ phương pháp nào không được đóng gói trong vòng lớp. Nó nên được truy cập thông qua các phương pháp setValues​​() và getValues().

**2.3 Truyền thông điệp**

Bất kỳ ứng dụng nào cũng yêu cầu một số đối tượng tương tác một cách hài hòa. Các đối tượng trong một hệ thống có thể giao tiếp với nhau bằng cách truyền thông điệp. Giả sử một hệ thống có hai đối tượng: obj1 và obj2. Đối tượng obj1 gửi một thông báo tới đối tượng obj2, nếu obj1 muốn obj2 thực hiện một trong các phương thức của nó.

Các tính năng truyền thông điệp là:

* Thông điệp đi qua giữa hai object nói chung là không theo chiều hướng.
* Cho phép tất cả các tương tác giữa các đối tượng.
* Chủ yếu liên quan đến việc gọi phương thức của lớp.
* Đối tượng trong các quá trình khác nhau có thể được tham gia vào việc truyền thông điệp.

**2.4 Kế thừa**

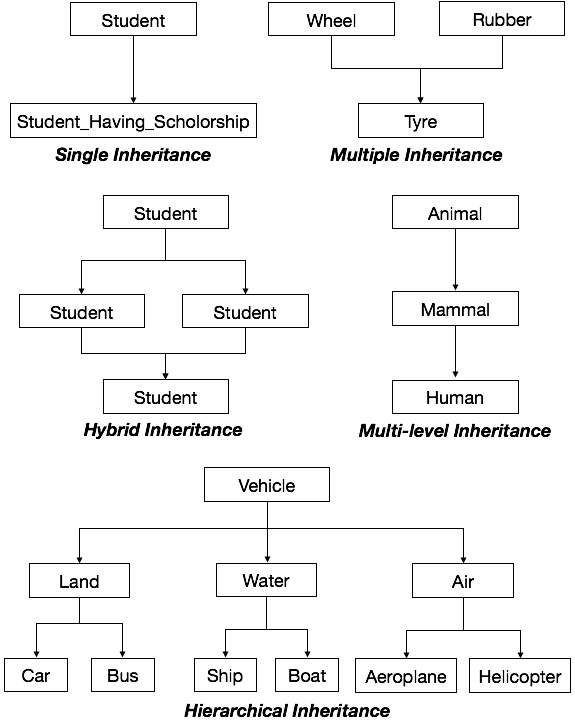
Kế thừa là cơ chế cho phép các lớp mới được tạo ra từ các lớp hiện có bằng cách mở rộng và tinh chỉnh các khả năng của nó. Các lớp hiện có được gọi là các lớp cơ sở / lớp cha / các super class, và các lớp mới được gọi là lớp con / subclass. Lớp con có thể kế thừa hoặc rút ra được các thuộc tính và phương thức của lớp cha miễn là lớp lớp cha cho phép như vậy. Bên cạnh đó, lớp con có thể thêm các thuộc tính và phương thức riêng của nó và có thể sửa đổi bất kỳ phương thức của lớp cha.

Thí dụ

Từ lớp Thú, một số lớp có thể được sinh ra như: Con người, Chó, Bò. Các lớp này đều có các đặc điểm riêng biệt của lớp thú. Ngoài ra, mỗi lớp đều có đặc điểm riêng. Có thể nói rằng một con bò là "một" động vật có vú.

Các loại kế thừa:

* Đơn kế thừa: Một phân lớp xuất phát từ một lớp cha duy nhất.
* Đa kế thừa: Một phân lớp có nguồn gốc từ nhiều hơn một lớp cha.
* Kế thừa đa tầng: Một phân lớp có nguồn gốc từ một lớp cha mà lớp cha này lại có nguồn gốc từ một lớp khác...
* Kế thừa cấp bậc: Một lớp có một số lớp con mà mỗi lớp có thể có các lớp con tiếp theo… để tạo thành một cấu trúc cây.
* Kế thừa hỗn hợp: Sự kết hợp đa thừa kế và đa tầng để tạo thành một cấu trúc lưới.



**2.5 Đa hình**

Đa hình ban đầu là một từ tiếng Hy Lạp có nghĩa là khả năng có nhiều hình thức. Trong mô hình hướng đối tượng, đa hình hàm ý sử dụng các hoạt động theo những cách khác nhau, tùy thuộc vào thể hiện mà chúng đang hoạt động. Đa hình cho phép các đối tượng có cấu trúc bên trong khác nhau có một giao diện bên ngoài thông thường. Đa hình có hiệu quả đặc biệt trong việc cài đặt các instance.

Thí dụ

Chúng ta hãy xem xét hai lớp, Circle và Square, mỗi một với một phương pháp findArea(). Mặc dù tên và mục đích của các phương thức trong các lớp là giống nhau nhưng phương thức tính diện tích là khác nhau cho mỗi lớp. Khi một đối tượng của lớp Circle gọi phương thức findArea() của nó, thao tác sẽ tìm diện tích của vòng tròn mà không có xung đột với phương thức findArea() của lớp Square.

**2.6 Tổng quát hóa và Chuyên biệt hóa**

Tổng quát hóa và chuyên môn hóa đại diện cho một hệ thống thứ bậc của các mối quan hệ giữa các lớp, nơi các lớp con kế thừa từ lớp cha.

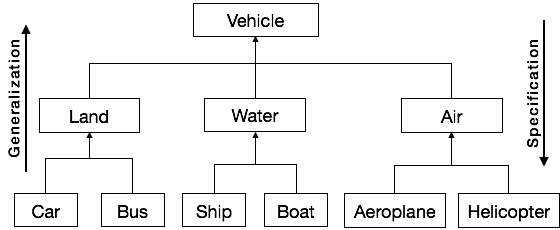
**2.6.1 Tổng quát hóa**

Trong quá trình tổng quát, các đặc tính chung của các lớp học được kết hợp để tạo thành một lớp ở cấp bậc cao hơn, nghĩa là các lớp con được kết hợp để tạo ra một lớp cha tổng quát. Nó đại diện cho một mối quan hệ "is-a-kind-of". Ví dụ: "xe hơi là một loại phương tiện đường bộ", hoặc "thuyền là một loại phương tiện đường thủy".

**2.6.2 Chuyên biệt hóa**

Chuyên biệt hoá là quá trình ngược lại của khái quát hóa. Ở đây, các tính năng phân biệt của các nhóm đối tượng được sử dụng để tạo ra các lớp chuyên biệt từ các lớp hiện có. Có thể nói rằng các lớp con là các phiên bản chuyên biệt của lớp cha.

Hình dưới đây cho thấy một ví dụ về khái quát hóa và chuyên biệt hóa.



**2.7 Kết nối (Link) và Liên kết (Association)**

**2.7.1 Kết nối**

Kết nối đại diện cho một quan hệ thông qua đó một đối tượng hợp tác với các đối tượng khác. Rumbaugh đã định nghĩa nó là "một kết nối vật lý hay khái niệm giữa các đối tượng". Thông qua kết nối, một đối tượng có thể gọi các phương thức hoặc điều hướng qua một đối tượng khác. Một kết nối mô tả mối quan hệ giữa hai hoặc nhiều đối tượng.

**2.7.2 Liên kết**

Liên kết là một nhóm kết nối có cấu trúc chung và hành vi chung. Liên kết miêu tả mối quan hệ giữa các đối tượng của một hoặc nhiều lớp. Một kết nối có thể được định nghĩa như một thể hiện của một liên kết.

**Cấp độ của liên kết**

Mức độ của một liên kết biểu thị số lớp học tham gia vào một kết nối. Các cấp độ có thể là đơn, đôi, hoặc nhiều.

* Liên kết đơn: kết nối các đối tượng của cùng một lớp.
* Liên kết đôi: kết nối các đối tượng của hai lớp.
* Đa liên kết: kết nối các đối tượng từ ba lớp trở lên.

**Kiểu liên kết**

3 kiểu liên kết:

* 1-1: Một đối tượng duy nhất của lớp A được kết hợp với một đối tượng duy nhất của lớp B.
* 1-n: Một đối tượng duy nhất của lớp A được kết hợp với nhiều đối tượng của lớp B.
* m-n: Một đối tượng của lớp A có thể được kết nối với nhiều đối tượng của lớp B và ngược lại một đối tượng của lớp B có thể được kết nối với nhiều đối tượng thuộc lớp A.

**2.8 Tổng hợp**

Tổng hợp là một mối quan hệ giữa các lớp theo đó lớp có thể được tạo thành từ bất kỳ sự kết hợp của các đối tượng của các lớp khác. Nó cho phép các đối tượng được đặt trực tiếp trong thân của các lớp khác. Tập hợp được gọi là mối quan hệ "một phần" hoặc "có-một", với khả năng điều hướng từ toàn bộ sang các phần của nó. Một đối tượng là tổng hợp của một đối tượng bao gồm một hoặc nhiều đối tượng khác.

Thí dụ

Trong mối quan hệ, "một chiếc ô tô có động cơ", xe là toàn bộ đối tượng và động cơ là một phần của chiếc xe. Tổng hợp có thể biểu thị:

* Vật lý: ví dụ, máy tính bao gồm màn hình, CPU, chuột, bàn phím,...
* Khái niệm: Ví dụ, cổ đông có - một cổ phần.

**2.9 Những lợi ích của việc sử dụng mô hình đối tượng là:**

* Giúp phát triển phần mềm nhanh hơn.
* Dễ dàng để duy trì. Giả sử một mô đun phát sinh lỗi, thì một lập trình viên có thể sửa mô đun đó, trong khi các phần khác của phần mềm vẫn đang chạy.
* Hỗ trợ nâng cấp tương đối phức tạp.
* Cho phép tái sử dụng các vật thể, thiết kế, và các chức năng.
* Làm giảm rủi ro phát triển, đặc biệt là trong việc tích hợp các hệ thống phức tạp.

## 3. Object Oriented System

**3.1 Các giai đoạn trong phát triển phần mềm hướng đối tượng**

Các giai đoạn chính của phát triển phần mềm sử dụng phương pháp hướng đối tượng là phân tích hướng đối tượng, thiết kế hướng đối tượng và thực hiện theo định hướng đối tượng.

**3.2 Phân tích hướng đối tượng**

Trong giai đoạn này, vấn đề được xây dựng, yêu cầu người dùng được xác định, và sau đó một mô hình được xây dựng dựa trên các đối tượng thực tế. Quá trình phân tích tạo ra các mô hình mà hệ thống mong muốn, và phải trả lời được câu hỏi nó sẽ hoạt động như thế nào và làm thế nào nó phải được phát triển. Các mô hình không bao gồm bất kỳ chi tiết thực hiện nào để nó có thể được hiểu và kiểm tra bởi bất kỳ một người nào không chuyên về kỹ thuật.

**3.3 Thiết kế hướng đối tượng**

Thiết kế hướng đối tượng bao gồm hai giai đoạn chính là thiết kế hệ thống và thiết kế đối tượng.

**3.3.1 THIẾT KẾ HỆ THỐNG**

Trong giai đoạn này, kiến ​​trúc hoàn chỉnh của hệ thống được thiết kế. Hệ thống được hình thành như một tập hợp các hệ thống con tương tác với nhau mà lần lượt bao gồm một thứ bậc của các đối tượng tương tác, được nhóm thành các lớp. Thiết kế hệ thống được thực hiện theo cả mô hình phân tích hệ thống và kiến ​​trúc hệ thống được đề xuất. Ở đây, trọng tâm sẽ là các đối tượng bao gồm hệ thống, không phải là quá trình trong hệ thống.

**3.3.2 THIẾT KẾ ĐỐi TƯỢNG**

Trong giai đoạn này, một mô hình thiết kế được phát triển dựa trên cả hai mô hình trong giai đoạn phân tích hệ thống và kiến ​​trúc được thiết kế trong giai đoạn thiết kế hệ thống. Tất cả các lớp mong muốn được xác định. Người thiết kế quyết định xem:

* Các lớp mới sẽ được tạo ra từ đầu
* Bất kỳ lớp hiện có nào có thể được sử dụng ở dạng ban đầu của chúng
* Các lớp mới nên được kế thừa từ các lớp hiện có.

Các mối liên kết giữa các lớp được thiết lập và phân cấp các lớp được xác định. Bên cạnh đó, nhà phát triển thiết kế các chi tiết nội bộ của các lớp và các mối liên kết giữa chúng, tức là cấu trúc dữ liệu cho mỗi thuộc tính và các thuật toán cho mỗi phương thức.

**3.4 Thực thi và Kiểm thử hướng đối tượng**

Trong giai đoạn này, mô hình thiết kế được thực thi bằng code. Cơ sở dữ liệu được tạo ra và yêu cầu phần cứng được xác định cụ thể. Một khi code, nó được kiểm thử bằng cách sử dụng các kỹ thuật chuyên biệt để xác định và loại bỏ các lỗi trong code.

**3.4.1 Nguyên tắc của các hệ thống hướng đối tượng**

Bốn yếu tố chính là:

* Trừu tượng
* Đóng gói
* Mô đun
* Cấp bậc

**Trừu tượng**

Việc trừu tượng có nghĩa là tập trung vào các tính năng thiết yếu của một phần tử hoặc đối tượng trong OOP, bỏ qua các thuộc tính không liên quan hoặc tình cờ của nó. Các tính năng thiết yếu là tương đối so với bối cảnh trong đó đối tượng đang được sử dụng.

Ví dụ: Khi một lớp Student được thiết kế, các thuộc tính như tên, khóa học và địa chỉ được tạo ra trong khi các thuộc tính như nhịp tim hay cỡ giày được loại bỏ, vì chúng không liên quan.

**Đóng gói**

Đóng gói là quá trình ràng buộc cả thuộc tính và phương thức với nhau trong một lớp. Thông qua việc đóng gói, các chi tiết nội bộ của một lớp học có thể được ẩn từ bên ngoài. Lớp này có các phương thức cung cấp tới giao diện người dùng bằng cách cung cấp các dịch vụ mà lớp cung cấp.

**Mô đun**

Mô đun là quá trình phân rã một chương trình thành một tập các mô đun để giảm sự phức tạp tổng thể của chương trình.

Mô đun liên kết với đóng gói. Mô đun có thể được hình dung như là một cách để lập bản đồ abstractions đóng gói thành các mô đun vật lý thật sự, có độ gắn kết cao trong các mô đun và sự tương tác giữa các mô-đun của chúng hoặc sự ghép nối là thấp.

**Phân cấp**

Hai loại phân cấp trong OOA là:

* "IS-A": Nó định nghĩa quan hệ phân cấp trong kế thừa, trong đó từ một lớp cha, một số lớp con có thể được lấy. Ví dụ, nếu chúng ta lấy được một lớp Rose từ một lớp Flower, chúng ta có thể nói rằng hoa hồng là một loại hoa.
* "PART-OF": Nó định nghĩa quan hệ phân cấp trong tập hợp theo đó lớp có thể bao gồm các lớp khác. Ví dụ, một bông hoa bao gồm cánh hoa, cánh hoa, nhị hoa. Có thể nói rằng một cánh hoa là một phần của hoa.

## 4. Object Oriented Analysis

Trong phân tích hệ thống hoặc phân tích hướng đối tượng, yêu cầu hệ thống được xác định, các lớp và các mối quan hệ giữa các lớp được xác định.

Ba kỹ thuật phân tích được sử dụng kết hợp với nhau để phân tích hướng đối tượng là mô hình đối tượng, mô hình động và mô hình chức năng.

**4.1 Mô hình đối tượng**

Mô hình hóa đối tượng phát triển cấu trúc tĩnh của hệ thống phần mềm dưới dạng các đối tượng. Nó xác định các đối tượng, các lớp mà trong đó các đối tượng có thể được nhóm vào và các mối quan hệ giữa các đối tượng. Nó cũng xác định các thuộc tính chính và các hoạt động đặc trưng cho mỗi lớp.

Quá trình mô hình đối tượng có thể được hình dung trong các bước sau:

* Xác định đối tượng và nhóm thành các lớp
* Xác định các mối quan hệ giữa các lớp
* Tạo sơ đồ mô hình đối tượng người dùng
* Xác định thuộc tính đối tượng người dùng
* Xác định các hoạt động cần được thực hiện trên các lớp
* Xem lại bảng thuật ngữ

**4.2 Mô hình hóa động**

Sau khi hành vi tĩnh của hệ thống được phân tích, hành vi của nó đối với thời gian và thay đổi bên ngoài cần phải được kiểm tra. Đây là mục đích của mô hình động.

Quá trình mô hình động có thể được hình dung trong các bước sau:

* Xác định trạng thái của từng đối tượng
* Xác định các sự kiện và phân tích khả năng áp dụng các hành động
* Xây dựng sơ đồ mô hình động, bao gồm sơ đồ chuyển đổi trạng thái
* Thể hiện mỗi trạng thái theo các thuộc tính của đối tượng
* Xác thực các sơ đồ chuyển trạng thái được rút ra

**4.3 Mô hình chức năng**

Mô hình chức năng là bước cuối cùng trong phân tích hướng đối tượng. Mô hình chức năng cho thấy các quá trình được thực hiện trong một đối tượng và cách dữ liệu thay đổi khi nó di chuyển giữa các phương thức. Nó xác định ý nghĩa của các hoạt động của mô hình đối tượng và các hành động của mô hình động. Mô hình chức năng tương ứng với sơ đồ luồng dữ liệu của phân tích cấu trúc truyền thống.

Quá trình mô hình hóa chức năng có thể được hình dung trong các bước sau:

* Xác định tất cả input và output
* Xây dựng sơ đồ luồng dữ liệu thể hiện các chức năng phụ thuộc
* Nêu rõ mục đích của mỗi chức năng
* Xác định những khó khăn
* Chỉ định các tiêu chí tối ưu hoá

## 5. Object Oriented Design

Sau giai đoạn phân tích, mô hình khái niệm được phát triển hơn nữa trong một mô hình hướng đối tượng sử dụng thiết kế hướng đối tượng (OOD). Trong OOD, các khái niệm không phụ thuộc vào công nghệ trong mô hình phân tích được ánh xạ vào việc thực thi các lớp, các ràng buộc được xác định, và các giao diện được thiết kế, tạo ra mô hình cho miền giải pháp. Tóm lại, mô tả chi tiết được xây dựng xác định cách thức hệ thống được xây dựng trên công nghệ cụ thể

Các giai đoạn thiết kế hướng đối tượng có thể được xác định là:

* Định nghĩa ngữ cảnh của hệ thống
* Thiết kế kiến ​​trúc hệ thống
* Xác định các đối tượng trong hệ thống
* Xây dựng mô hình thiết kế
* Đặc tả giao diện đối tượng

**5.1 Thiết kế hệ thống hướng đối tượng**

**5.1.1 Thiết kế hệ thống**

Thiết kế hệ thống hướng đối tượng liên quan đến việc xác định ngữ cảnh của một hệ thống theo sau bởi kiến ​​trúc hệ thống.

* **Bối cảnh:** Bối cảnh của một hệ thống có phần tĩnh và phần động. Bối cảnh tĩnh của hệ thống được thiết kế bằng cách sử dụng một sơ đồ khối đơn giản trên toàn bộ hệ thống được mở rộng thành một hệ thống phân cấp của các hệ thống con. Mô hình hệ thống con được đại diện bởi các gói UML. Bối cảnh động mô tả cách hệ thống tương tác với môi trường. Nó được mô hình bằng cách sử dụng sơ đồ use case.
* **Kiến trúc hệ thống:** Kiến trúc hệ thống được thiết kế dựa trên bối cảnh của hệ thống phù hợp với các nguyên tắc thiết kế kiến ​​trúc cũng như kiến ​​thức miền. Thông thường, một hệ thống được phân chia thành các lớp và mỗi lớp được phân rã để tạo thành các hệ thống con.

**5.1.2 Phân rã hướng đối tượng**

Sự phân rã có nghĩa là phân chia một hệ thống phức tạp lớn thành các thành phần nhỏ hơn với sự phức tạp thấp hơn, dựa trên các nguyên tắc chia để trị. Mỗi thành phần chính của hệ thống được gọi là một hệ thống con. Phân rã theo hướng đối tượng xác định các đối tượng tự trị cá nhân trong một hệ thống và sự giao tiếp giữa các đối tượng này.

Ưu điểm của sự phân rã là:

* Các thành phần riêng biệt ít phức tạp hơn, dễ hiểu hơn và dễ quản lý hơn.
* Nó cho phép phân công lực lượng lao động có kỹ năng chuyên môn.
* Nó cho phép các hệ thống con được thay thế hoặc sửa đổi mà không ảnh hưởng đến các hệ thống con khác.

**5.1.3 Xác định tính đồng thời**

Đồng thời cho phép nhiều đối tượng nhận các sự kiện cùng một lúc và nhiều hoạt động sẽ được thực hiện đồng thời. Đồng thời được xác định và đại diện trong mô hình động.

Để kích hoạt tính đồng thời, mỗi phần tử đồng thời được gán một luồng điều khiển riêng biệt. Nếu đồng thời là ở cấp độ đối tượng, sau đó hai đối tượng đồng thời được gán hai chuỗi điều khiển khác nhau. Nếu hai hoạt động của một đối tượng duy nhất là đồng thời, thì đối tượng đó được phân chia thực hiện ở các luồng khác nhau.

Truy cập đồng thời được kết hợp với các vấn đề về toàn vẹn dữ liệu, deadlock, and starvation. Vì vậy, một chiến lược rõ ràng cần phải được thực hiện bất cứ khi nào xử lý song song được yêu cầu. Bên cạnh đó, đồng thời cần phải tự xác định ở giai đoạn thiết kế, và không thể để lại cho giai đoạn thực hiện.

**5.1.4 Xác định các mẫu**

Trong khi thiết kế các ứng dụng, một số giải pháp được áp dụng cho một số loại vấn đề. Đây là những mẫu thiết kế. Một mẫu có thể được định nghĩa như là một bộ các khối xây dựng có thể được sử dụng trong một số loại vấn đề nhất định khi phát triển ứng dụng.

Một số mẫu thiết kế thường sử dụng là:

* Façade
* Model view separation
* Observer
* MVC
* Publish subscribe
* Proxy

**5.1.5 Kiểm soát sự kiện**

Trong quá trình thiết kế hệ thống, các sự kiện xảy ra trong các đối tượng của hệ thống cần được xác định và xử lý phù hợp.

Một sự kiện được xác đinh trong thời gian và không gian nhất định.

Có bốn loại sự kiện có thể được mô hình hóa, cụ thể là:

* Sự kiện tín hiệu: Một đối tượng có tên ném ra bởi một đối tượng và bị bắt bởi một đối tượng khác.
* Sự kiện gọi: Sự kiện đồng bộ đại diện cho vận chuyển của một hoạt động.
* Sự kiện thời gian: Một sự kiện đại diện cho thời gian.
* Sự kiện thay đổi: Sự kiện thể hiện sự thay đổi trạng thái.

**5.1.6 Xử lý các điều kiện ranh giới**

Giai đoạn thiết kế hệ thống cần giải quyết việc khởi tạo và chấm dứt hệ thống như một toàn bộ cũng như ở từng hệ thống con. Các khía cạnh khác nhau được ghi lại như sau:

* Sự khởi đầu của hệ thống, vd: sự chuyển đổi của hệ thống từ trạng thái chưa được khởi tạo sang trạng thái ổn định.
* Sự chấm dứt của hệ thống, vd: đóng tất cả các luồng đang chạy, dọn dẹp các tài nguyên, và các tin nhắn đã được gửi.
* Cấu hình ban đầu của hệ thống và cấu hình lại hệ thống khi cần thiết.
* Dự báo sự cố không mong muốn của hệ thống.

Các điều kiện ranh giới được mô hình bằng các use case.

**5.2 Thiết kế đối tượng**

Sau khi hệ thống phân cấp được phát triển, các đối tượng trong hệ thống được xác định và các chi tiết được thiết kế. Ở đây, nhà thiết kế chi tiết về chiến lược được lựa chọn trong suốt quá trình thiết kế hệ thống. Những thứ quan trọng chuyển từ khái niệm miền ứng dụng sang các khái niệm máy tính. Các đối tượng được xác định trong khi phân tích được khắc phục để thực hiện với mục tiêu giảm thiểu thời gian, bộ nhớ và tổng chi phí.

Thiết kế đối tượng bao gồm các giai đoạn sau:

* Nhận dạng đối tượng
* Biểu diễn đối tượng, tức là xây dựng các mô hình thiết kế
* Phân loại hoạt động
* Thiết kế thuật toán
* Thiết kế các quan hệ
* Kiểm soát các tương tác bên ngoài
* Đóng gói lớp và liên kết thành các mô-đun

**5.2.1 Xác định đối tượng**

Bước đầu tiên của thiết kế đối tượng là xác định đối tượng. Các đối tượng được xác định trong khi phân tích được nhóm lại thành các lớp và được tối ưu sao cho phù hợp với thực hiện thực tế.

Các chức năng của giai đoạn này là:

* Xác định và tối ưu các lớp trong mỗi hệ thống con hoặc gói
* Xác định liên kết giữa các lớp
* Thiết kế các mối liên kết phân cấp giữa các lớp
* Thiết kế tổng hợp

**5.2.2 Đại diện đối tượng**

Khi các lớp được xác định, chúng cần được biểu diễn bằng các kỹ thuật mô hình đối tượng. Giai đoạn này chủ yếu liên quan đến việc xây dựng sơ đồ UML.

Có hai loại mô hình thiết kế:

* Mô hình tĩnh: Để mô tả cấu trúc tĩnh của một hệ thống sử dụng sơ đồ lớp và sơ đồ đối tượng.
* Mô hình động: Để mô tả cấu trúc động của một hệ thống và cho thấy sự tương tác giữa các lớp bằng cách sử dụng sơ đồ tương tác và sơ đồ biểu đồ trạng thái.

**5.2.3 Phân loại hoạt động**

Trong bước này, hoạt động thực hiện trên các đối tượng được xác định bằng cách kết hợp 3 mô hình phát triển trong giai đoạn OOA, cụ thể là mô hình đối tượng, mô hình động và mô hình chức năng. Một hoạt động sẽ chỉ định cách thực hiện.

Các nhiệm vụ sau được thực hiện liên quan đến hoạt động:

* Phát triển sơ đồ chuyển đổi trạng thái của từng đối tượng trong hệ thống được.
* Xác định các hoạt động, sự kiện.
* Xác định các trường hợp trong đó một sự kiện của cùng một hoặc nhiều đối tượng được kích
* Xác định các hoạt động phụ.
* Mở rộng các hành động chính cho sơ đồ luồng dữ liệu.

**5.2.3 Thiết kế thuật toán**

Các hoạt động trong các đối tượng được xác định bằng cách sử dụng các thuật toán. Một thuật toán là một thủ tục để giải quyết vấn đề đặt ra trong một thao tác. Các thuật toán tập trung vào cách nó được thực hiện.

Có thể có nhiều hơn một thuật toán tương ứng với một hoạt động nhất định. Khi các thuật toán được xác định, thuật toán tối ưu được chọn vào miền vấn đề đã cho. Các số liệu để chọn thuật toán tối ưu là:

* Độ phức tạp trong tính toán: xác định độ hiệu quả của một thuật toán về thời gian tính toán và yêu cầu về bộ nhớ.
* Tính linh hoạt: xác định xem các thuật toán đã chọn có thể được thực hiện phù hợp, không mất sự phù hợp trong các môi trường khác nhau.
* Khả năng có thể hiểu được: Điều này quyết định xem thuật toán đã chọn có dễ để cài đặt hay không.

**5.3.4 Thiết kế mối quan hệ**

Chiến lược để cài đặt quan hệ cần phải được phác hoạ ra trong giai đoạn thiết kế đối tượng. Các mối quan hệ chính được giải quyết bao gồm các liên kết, tập hợp và kế thừa.

Nhà thiết kế nên làm những điều sau:

* Xác định liệu một liên kết có mang tính đơn hướng hay đa hướng.
* Phân tích liên kết và cập nhật chúng nếu cần.
* Thực hiện các liên kết như một đối tượng riêng biệt

Về thừa kế, người thiết kế nên làm như sau:

* Điều chỉnh các lớp và sự liên kết giữa chúng.
* Xác định các lớp trừu tượng.
* Thực hiện các quy định để các hành vi được chia sẻ khi cần thiết.

**5.3.5 Cài đặt điều khiển**

Trong thiết kế hệ thống, một chiến lược cơ bản để thực hiện mô hình động được thực hiện. Trong quá trình thiết kế đối tượng, chiến lược này phù hợp với việc triển khai thực hiện.