UML và các giai đoạn của chu trình phát triển phần mềm

Bởi:
duongkieuhoa
tonthathoaan

UML và các giai đoạn cỦa chu trình phát triển phần mềm

Giai đoạn nghiên cứu sơ bộ:

UML đưa ra khái niệm **Use Case** để nắm bắt các yêu cầu của khách hàng (người sử dụng). UML sử dụng biểu đồ Use case (Use Case Diagram) để nêu bật mối quan hệ cũng như sự giao tiếp với hệ thống.

Qua phương pháp mô hình hóa Use case, các tác nhân (Actor) bên ngoài quan tâm đến hệ thống sẽ được mô hình hóa song song với chức năng mà họ đòi hỏi từ phía hệ thống (tức là Use case). Các tác nhân và các Use case được mô hình hóa cùng các mối quan hệ và được miêu tả trong biểu đồ Use case của UML. Mỗi một Use case được mô tả trong tài liệu, và nó sẽ đặc tả các yêu cầu của khách hàng: Anh ta hay chị ta chờ đợi điều gì ở phía hệ thống mà không hề để ý đến việc chức năng này sẽ được thực thi ra sao.

Giai đoạn phân tích:

Giai đoạn phân tích quan tâm đến quá trình trừu tượng hóa đầu tiên (các lớp và các đối tượng) cũng như cơ chế hiện hữu trong phạm vi vấn đề. Sau khi nhà phân tích đã nhận biết được các lớp thành phần của mô hình cũng như mối quan hệ giữa chúng với nhau, các lớp cùng các mối quan hệ đó sẽ được miêu tả bằng công cụ biểu đồ lớp (class diagram) của UML. Sự cộng tác giữa các lớp nhằm thực hiện các Use case cũng sẽ được miêu tả nhờ vào các mô hình động (dynamic models) của UML. Trong giai đoạn phân tích, chỉ duy nhất các lớp có tồn tại trong phạm vi vấn đề (các khái niệm đời thực) là được mô hình hóa. Các lớp kỹ thuật định nghĩa chi tiết cũng như giải pháp trong hệ thống phần mềm, ví dụ như các lớp cho giao diện người dùng, cho ngân hàng dữ liệu, cho sự giao tiếp, trùng hợp, v.v..., chưa phải là mối quan tâm của giai đoạn này.

Giai đoạn thiết kế:

Trong giai đoạn này, kết quả của giai đoạn phân tích sẽ được mở rộng thành một giải pháp kỹ thuật. Các lớp mới sẽ được bổ sung để tạo thành một hạ tầng cơ sở kỹ thuật: Giao diện người dùng, các chức năng để lưu trữ các đối tượng trong ngân hàng dữ liệu, giao tiếp với các hệ thống khác, giao diện với các thiết bị ngoại vi và các máy móc khác trong hệ thống, ... Các lớp thuộc phạm vi vấn đề có từ giai đoạn phân tích sẽ được "nhúng" vào hạ tầng cơ sở kỹ thuật này, tạo ra khả năng thay đổi trong cả hai phương diện: Phạm vi vấn đề và hạ tầng cơ sở. Giai đoạn thiết kế sẽ đưa ra kết quả là bản đặc tả chi tiết cho giai đoạn xây dựng hệ thống.

Giai đoạn xây dựng:

Trong giai đoạn xây dựng (giai đoạn lập trình), các lớp của giai đoạn thiết kế sẽ được biến thành những dòng code cụ thể trong một ngôn ngữ lập trình hướng đối tượng cụ thể (không nên dùng một ngôn ngữ lập trình hướng chức năng!). Phụ thuộc vào khả năng của ngôn ngữ được sử dụng, đây có thể là một công việc khó khăn hay dễ dàng. Khi tạo ra các mô hình phân tích và thiết kế trong UML, tốt nhất nên cố gắng né tránh việc ngay lập tức biến đổi các mô hình này thành các dòng code. Trong những giai đoạn trước, mô hình được sử dụng để dễ hiểu, dễ giao tiếp và tạo nên cấu trúc của hệ thống; vì vậy, vội vàng đưa ra những kết luận về việc viết code có thể sẽ thành một trở ngại cho việc tạo ra các mô hình chính xác và đơn giản. Giai đoạn xây dựng là một giai đoạn riêng biệt, nơi các mô hình được chuyển thành code.

Thử nghiệm:

Như đã trình bày trong phần Chu Trình Phát Triển Phần Mềm, một hệ thống phần mềm thường được thử nghiệm qua nhiều giai đoạn và với nhiều nhóm thử nghiệm khác nhau. Các nhóm sử dụng nhiều loại biểu đồ UML khác nhau làm nền tảng cho công việc của mình: Thử nghiệm đơn vị sử dụng biểu đồ lớp (class diagram) và đặc tả lớp, thử nghiệm tích hợp thường sử dụng biểu đồ thành phần (component diagram) và biểu đồ cộng tác (collaboration diagram), và giai đoạn thử nghiệm hệ thống sử dụng biểu đồ Use case (use case diagram) để đảm bảo hệ thống có phương thức hoạt động đúng như đã được định nghĩa từ ban đầu trong các biểu đồ này.

Các thành phần của ngôn ngữ UML

Ngôn ngữ UML bao gồm một loạt các phần tử đồ họa (graphic element) có thể được kếp hợp với nhau để tạo ra các biểu đồ. Bởi đây là một ngôn ngữ, nên UML cũng có các nguyên tắc để kết hợp các phần tử đó.

Một số những thành phần chủ yếu của ngôn ngữ UML:

Hướng nhìn (view): Hướng nhìn chỉ ra những khía cạnh khác nhau của hệ thống cần phải được mô hình hóa. Một hướng nhìn không phải là một bản vẽ, mà là một sự trừu tượng hóa bao gồm một loạt các biểu đồ khác nhau. Chỉ qua việc định nghĩa của một loạt các hướng nhìn khác nhau, mỗi hướng nhìn chỉ ra một khía cạnh riêng biệt của hệ thống, người ta mới có thể tạo dựng nên một bức tranh hoàn thiện về hệ thống. Cũng chính các hướng nhìn này nối kết ngôn ngữ mô hình hóa với quy trình được chọn cho giai đoạn phát triển.

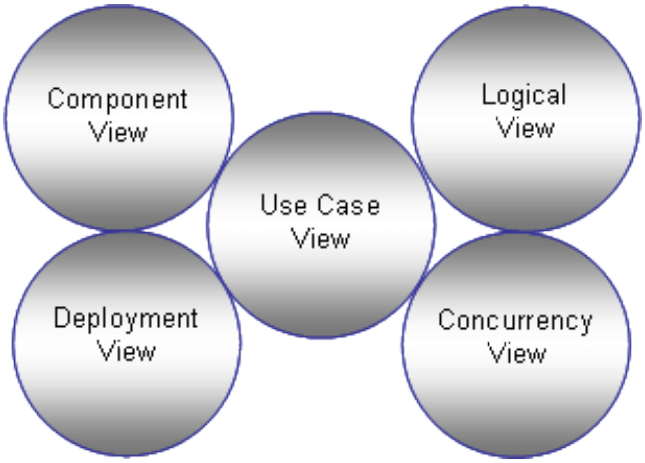
Biểu đồ (diagram): Biểu đồ là các hình vẽ miêu tả nội dung trong một hướng nhìn. UML có tất cả 9 loại biểu đồ khác nhau được sử dụng trong những sự kết hợp khác nhau để cung cấp tất cả các hướng nhìn của một hệ thống.

Phần tử mô hình hóa (model element): Các khái niệm được sử dụng trong các biểu đồ được gọi là các phần tử mô hình, thể hiện các khái niệm hướng đối tượng quen thuộc. Ví dụ như lớp, đối tượng, thông điệp cũng như các quan hệ giữa các khái niệm này, bao gồm cả liên kết, phụ thuộc, khái quát hóa. Một phần tử mô hình thường được sử dụng trong nhiều biểu đồ khác nhau, nhưng nó luôn luôn có chỉ một ý nghĩa và một kí hiệu.

Cơ chế chung: Cơ chế chung cung cấp thêm những lời nhận xét bổ sung, các thông tin cũng như các quy tắc ngữ pháp chung về một phần tử mô hình; chúng còn cung cấp thêm các cơ chế để có thể mở rộng ngôn ngữ UML cho phù hợp với một phương pháp xác định (một quy trình, một tổ chức hoặc một người dùng).

Hướng nhìn (View)

Mô hình hóa một hệ thống phức tạp là một việc làm khó khăn. Lý tưởng nhất là toàn bộ hệ thống được miêu tả chỉ trong một bản vẽ, một bản vẽ định nghĩa một cách rõ ràng và mạch lạc toàn bộ hệ thống, một bản vẽ ngoài ra lại còn dễ giao tiếp và dễ hiểu. Mặc dù vậy, thường thì đây là chuyện bất khả thi. Một bản vẽ không thể nắm bắt tất cả các thông tin cần thiết để miêu tả một hệ thống. Một hệ thống cần phải được miêu tả với một loạt các khía cạnh khác nhau: Về mặt chức năng (cấu trúc tĩnh của nó cũng như các tương tác động), về mặt phi chức năng (yêu cầu về thời gian, về độ đáng tin cậy, về quá trình thực thi, v.v. và v.v.) cũng như về khía cạnh tổ chức (tổ chức làm việc, ánh xạ nó vào các code module, ...). Vì vậy một hệ thống thường được miêu tả trong một loạt các hướng nhìn khác nhau, mỗi hướng nhìn sẽ thể hiện một bức ảnh ánh xạ của toàn bộ hệ thống và chỉ ra một khía cạnh riêng của hệ thống.



Hình 3.1- Các View trong UML

Mỗi một hướng nhìn được miêu tả trong một loạt các biểu đồ, chứa đựng các thông tin nêu bật khía cạnh đặc biệt đó của hệ thống. Trong thực tế khi phân tích và thiết kế rất dễ xảy ra sự trùng lặp thông tin, cho nên một biểu đồ trên thật tế có thể là thành phần của nhiều hướng nhìn khác nhau. Khi nhìn hệ thống từ nhiều hướng nhìn khác nhau, tại một thời điểm có thể người ta chỉ tập trung vào một khía cạnh của hệ thống. Một biểu đồ trong một hướng nhìn cụ thể nào đó cần phải đủ độ đơn giản để tạo điều kiện giao tiếp dễ dàng, để dính liền với các biểu đồ khác cũng như các hướng nhìn khác, làm sao cho bức tranh toàn cảnh của hệ thống được miêu tả bằng sự kết hợp tất cả các thông tin từ tất cả các hướng nhìn. Một biểu đồ chứa các kí hiệu hình học mô tả các phần tử mô hình của hệ thống. UML có tất cả các hướng nhìn sau:

- Hướng nhìn Use case (use case view) : đây là hướng nhìn chỉ ra khía cạnh chức năng của một hệ thống, nhìn từ hướng tác nhân bên ngoài.

- Hướng nhìn logic (logical view): chỉ ra chức năng sẽ được thiết kế bên trong hệ thống như thế nào, qua các khái niệm về cấu trúc tĩnh cũng như ứng xử động của hệ thống.

- Hướng nhìn thành phần (component view): chỉ ra khía cạnh tổ chức của các thành phần code.

- Hướng nhìn song song (concurrency view): chỉ ra sự tồn tại song song/ trùng hợp trong hệ thống, hướng đến vấn đề giao tiếp và đồng bộ hóa trong hệ thống.

- Hướng nhìn triển khai (deployment view): chỉ ra khía cạnh triển khai hệ thống vào các kiến trúc vật lý (các máy tính hay trang thiết bị được coi là trạm công tác).

Khi bạn chọn công cụ để vẽ biểu đồ, hãy chọn công cụ nào tạo điều kiện dễ dàng chuyển từ hướng nhìn này sang hướng nhìn khác. Ngoài ra, cho mục đích quan sát một chức năng sẽ được thiết kế như thế nào, công cụ này cũng phải tạo điều kiện dễ dàng cho bạn

chuyển sang hướng nhìn Use case (để xem chức năng này được miêu tả như thế nào từ phía tác nhân), hoặc chuyển sang hướng nhìn triển khai (để xem chức năng này sẽ được phân bố ra sao trong cấu trúc vật lý - Nói một cách khác là nó có thể nằm trong máy tính nào).

Ngoài các hướng nhìn kể trên, ngành công nghiệp phần mềm còn sử dụng cả các hướng nhìn khác, ví dụ hướng nhìn tĩnh-động, hướng nhìn logic-vật lý, quy trình nghiệp vụ (workflow) và các hướng nhìn khác. UML không yêu cầu chúng ta phải sử dụng các hướng nhìn này, nhưng đây cũng chính là những hướng nhìn mà các nhà thiết kế của UML đã nghĩ tới, nên có khả năng nhiều công cụ sẽ dựa trên các hướng nhìn đó.

Hướng nhìn Use case (Use case View):

Hướng nhìn Use case miêu tả chức năng của hệ thống sẽ phải cung cấp do được tác nhân từ bên ngoài mong đợi. Tác nhân là thực thể tương tác với hệ thống; đó có thể là một người sử dụng hoặc là một hệ thống khác. Hướng nhìn Use case là hướng nhìn dành cho khách hàng, nhà thiết kế, nhà phát triển và người thử nghiệm; nó được miêu tả qua các biểu đồ Use case (use case diagram) và thỉnh thoảng cũng bao gồm cả các biểu đồ hoạt động (activity diagram). Cách sử dụng hệ thống nhìn chung sẽ được miêu tả qua một loạt các Use case trong hướng nhìn Use case, nơi mỗi một Use case là một lời miêu tả mang tính đặc thù cho một tính năng của hệ thống (có nghĩa là một chức năng được mong đợi).

Hướng nhìn Use case mang tính trung tâm, bởi nó đặt ra nội dung thúc đẩy sự phát triển các hướng nhìn khác. Mục tiêu chung của hệ thống là cung cấp các chức năng miêu tả trong hướng nhìn này – cùng với một vài các thuộc tính mang tính phi chức năng khác – vì thế hướng nhìn này có ảnh hưởng đến tất cả các hướng nhìn khác. Hướng nhìn này cũng được sử dụng để thẩm tra (verify) hệ thống qua việc thử nghiệm xem hướng nhìn Use case có đúng với mong đợi của khách hàng (Hỏi: "Đây có phải là thứ bạn muốn") cũng như có đúng với hệ thống vừa được hoàn thành (Hỏi: "Hệ thống có hoạt động như đã đặc tả?").

Hướng nhìn logic (Logical View):

Hướng nhìn logic miêu tả phương thức mà các chức năng của hệ thống sẽ được cung cấp. Chủ yếu nó được sử dụng cho các nhà thiết kế và nhà phát triển. Ngược lại với hướng nhìn Use case, hướng nhìn logic nhìn vào phía bên trong của hệ thống. Nó miêu tả kể cả cấu trúc tĩnh (lớp, đối tượng, và quan hệ) cũng như sự tương tác động sẽ xảy ra khi các đối tượng gửi thông điệp cho nhau để cung cấp chức năng đã định sẵn. Hướng nhìn logic định nghĩa các thuộc tính như trường tồn (persistency) hoặc song song (concurrency), cũng như các giao diện cũng như cấu trúc nội tại của các lớp.

Cấu trúc tĩnh được miêu tả bằng các biểu đồ lớp (class diagram) và biểu đồ đối tượng (object diagram). Quá trình mô hình hóa động được miêu tả trong các biểu đồ trạng thái (state diagram), biểu đồ trình tự (sequence diagram), biểu đồ tương tác (collaboration diagram) và biểu đồ hoạt động (activity diagram).

Hướng nhìn thành phần (Component View):

Là một lời miêu tả của việc thực thi các modul cũng như sự phụ thuộc giữa chúng với nhau. Nó thường được sử dụng cho nhà phát triển và thường bao gồm nhiều biểu đồ thành phần. Thành phần ở đây là các modul lệnh thuộc nhiều loại khác nhau, sẽ được chỉ ra trong biểu đồ cùng với cấu trúc cũng như sự phụ thuộc của chúng. Các thông tin bổ sung về các thành phần, ví dụ như vị trí của tài nguyên (trách nhiệm đối với một thành phần), hoặc các thông tin quản trị khác, ví dụ như một bản báo cáo về tiến trình của công việc cũng có thể được bổ sung vào đây.

Hướng nhìn song song (Concurrency View):

Hướng nhìn song song nhắm tới sự chia hệ thống thành các qui trình (process) và các bộ xử lý (processor). Khía cạnh này, vốn là một thuộc tính phi chức năng của hệ thống, cho phép chúng ta sử dụng một cách hữu hiệu các nguồn tài nguyên, thực thi song song, cũng như xử lý các sự kiện không đồng bộ từ môi trường. Bên cạnh việc chia hệ thống thành các tiểu trình có thể được thực thi song song, hướng nhìn này cũng phải quan tâm đến vấn đề giao tiếp và đồng bộ hóa các tiểu trình đó.

Hướng nhìn song song giành cho nhà phát triển và người tích hợp hệ thống, nó bao gồm các biểu đồ động (trạng thái, trình tự, tương tác và hoạt động) cùng các biểu đồ thực thi (biểu đồ thành phần và biểu đồ triển khai).

Hướng nhìn triển khai (Deployment View):

Cuối cùng, hướng nhìn triển khai chỉ cho chúng ta sơ đồ triển khai về mặt vật lý của hệ thống, ví dụ như các máy tính cũng như các máy móc và sự liên kết giữa chúng với nhau. Hướng nhìn triển khai giành cho các nhà phát triển, người tích hợp cũng như người thử nghiệm hệ thống và được thể hiện bằng các biểu đồ triển khai. Hướng nhìn này cũng bao gồm sự ánh xạ các thành phần của hệ thống vào cấu trúc vật lý; ví dụ như chương trình nào hay đối tượng nào sẽ được thực thi trên máy tính nào.