Chương 1: TỔNG QUAN VỀ CÔNG NGHỆ PHẦN MỀM

1. CÁC KHÁI NIỆM CƠ BẢN

1.1. Phần mềm

1.1.1. Các khái niệm

Chương trình máy tính là một trình tự các chỉ thị để hướng dẫn máy tính làm việc nhằm hoàn thành một công việc nào đó do con người yêu cầu.

Phần mềm là một hệ thống các chương trình có thể thực hiện trên máy tính nhằm hỗ trợ các nhà chuyên môn trong từng lĩnh vực chuyên ngành thực hiện tốt nhất các thao tác nghiệp vụ của mình. Nhiệm vụ chính yếu của phần mềm là cho phép các nhà chuyên môn thực hiện các công việc của họ trên máy tính dễ dàng và nhanh chóng hơn so với khi thực hiện cùng công việc đó trong thế giới thực.

Hoạt động của mọi phần mềm là sự mô phỏng lại các hoạt động của thế giới thực trong một góc độ thu hẹp nào đó trên máy tính. Quá trình sử dụng một phần mềm chính là quá trình người dùng thực hiện các công việc trên máy tính để hoàn tất một công việc tương đương trong thế giới thực.

Lớp phần mềm là hệ thống các phần mềm trên cùng lĩnh vực hoạt động nào đó. Do cùng lĩnh vực hoạt động nên các phần mềm này thường có cấu trúc và chức năng (công việc mà người dùng thực hiện trên máy tính) tương tự nhau. Mục tiêu của ngành công nghệ phần mềm là hướng đến không những xây dựng được các phần mềm có chất lượng mà còn cho phép

xây dựng dễ dàng một phần mềm mới từ các phần mềm đã có sẵn trong cùng kĩnh vực (thậm chí trong các lĩnh vực khác).

STT	Lớp phần mềm	Các phần mềm
1	Hỗ trợ giải bài tập	lượng giác, hình học, giải tích, số học, …
2	Trò chơi	cờ carô, cờ tướng, cờ vua, xếp hình, …
3	Xếp lịch biểu	thi đấu, thời khóa biểu, hội nghị, …
4	Xét tuyển	nhân sự, học sinh lớp 10…
5	Bình chọn	Sản phẩm, cầu thủ, …
6	Quản lý học sinh	Mầm non, trung học, trung tâm…
7	Bán hàng	thuốc tây, vật liệu xây dựng, máy tính
8	Quản lý thuê bao	điện, điện thoại, nước, …
9	Cho mượn	sách, truyện, phim, …

***Bảng 1.1**: Các phần mềm và lớp phần mềm tương ứng*

1.1.2. Phân loại

Phần mềm hệ thống là những phần mềm đảm nhận công việc tích hợp và điều khiển các thiết bị phần cứng đồng thời tạo ra môi trường thuận lợi để các phần mềm khác và người sử dụng có thể thao tác trên đó như một khối thống nhất mà không cần phải quan tâm đến những chi tiết kỹ thuật phức tạp bên dưới như cách thức trao đổi dữ liệu giữa bộ nhớ chính và đĩa, cách hiển thị văn bản lên màn hình, ...

Phần mềm ứng dụng là những phần mềm được dùng để thực hiện một công việc xác định nào đó. Phần mềm ứng dụng có thể chỉ gồm một chương trình đơn giản như chương trình

xem ảnh, hoặc một nhóm các chương trình cùng tương tác với nhau để thực hiện một công việc nào đó như chương trình xử lý bản tính, chương trình xử lý văn bản, ...

1.1.3. Kiến trúc phần mềm

Sau khi đã có các khái niêm cơ bản nhất về phần mềm, tiếp sau đây chúng ta sẽ đi sâu vào tìm hiểu cấu trúc chi tiết các cấu trúc chi tiết các thành phần bên trong phần mềm. Phần mềm bao gồm 3 thành phần:

a) Thành phần giao tiếp (giao diện)

Cho phép tiếp nhận các yêu cầu về việc muốn thực hiện và cung cấp các dữ liệu nguồn liên quan đến công việc đó hoặc từ các thiết bị thu thập dữ liệu (cân, đo nhiệt độ, tế bào quang học, …)

Cho phép trình bày các kết quả của việc thực hiện các yêu cầu cho người dùng (kết quả của công việc khi thực hiện trên máy tính) hoặc điều khiển hoạt động các thiết bị điều khiển (đóng mở cửa, bật mở máy…)

Một cách tổng quát thành phần giao tiếp là hệ thống các hàm chuyên về việc nhập/xuất dữ liệu (hàm nhập/xuất) cùng với hình thức trình bày và tổ chức lưu trữ dữ liệu tương ứng, mục tiêu chính của các hàm này là đưa dữ liệu từ thế giới bên ngoài phần mềm vào bên trong hoặc ngược lại.

Trong phạm vi giáo trình này chỉ giới hạn xét đến giao tiếp với người sử dụng phần mềm và khi đó có tên gọi cụ thể hơn là thành phần giao diện.

b) Thành phần dữ liệu

Cho phép lưu trữ lại (hàm ghi) các kết quả đã xử lý (việc mượn sách đã được kiểm tra hợp lệ, bảng lương tháng đã được

tính) trên bộ nhớ phụ với tổ chức lưu trữ được xác định trước (tập tin có cấu trúc, tập tin nhị phân, cơ sở dữ liệu).

Cho phép truy xuất lại (hàm đọc) các dữ liệu đã lưu trữ phục vụ cho các hàm xử lý tương ứng.

Một cách tổng quát thành phần dữ liệu là hệ thống các hàm chuyên về đọc ghi dữ liệu (hàm đọc/ghi) cùng với mô hình tổ chức dữ liệu tương ứng. Mục tiêu chính của các hàm này là chuyển đổi dữ liệu giữa bộ nhớ chính và bộ nhớ phụ.

c) Thành phần xử lý

Kiểm tra tính hợp lệ của các dữ liệu nguồn được cung cấp từ người dùng theo các quy trình ràng buộc trong thế giới thực (chỉ cho mượn tối đa 3 quyển sách, mỗi lớp học có tối đa 50 học sinh, …)

Tiến hành xử lý cho ra kết quả mong đợi theo quy định tính toán có sẵn trong thế giới thực (quy tắc tính tiền phạt khi trả sách trễ, quy tắc tính tiền điện, quy tắc trả góp khi mua nhà…) hoặc theo thuật giải tự đề xuất (xếp thời khóa biểu tự động, nén ảnh…)

Việc xử lý dựa trên dữ liệu nguồn từ người sử dụng cung cấp (tính nghiệm phương trình bậc 2 dựa trên các hệ số đã nhập) hoặc dữ liệu lưu trữ đã có sẵn (tính tồn kho tháng dựa trên các phiếu nhập xuất đã lưu trữ…) hoặc cả hai (tính tiền phạt dựa trên ngày trả sách được nhập vào và thông tin về loại sách đã được lưu trữ…) tùy vào xử lý cụ thể. Tương tự, việc xử lý cho ra kết quả có thể dùng để xuất cho người dùng xem qua thành phần giao diện (trình bày nghiệm, xuất tiền phạt), hay cùng có thể lưu trữ lại qua thành phần dữ liệu (sổ sách

hiện đang được mượn của một độc giả…) hoặc cả hai (bảng lương, bảng tồn kho…)

Một cách tổng quát, thành phần xử lý là hệ thống các hàm chuyên về xử lý tính toán, biến đổi dữ liệu. Các hàm này sẽ dùng dữ liệu nguồn từ các hàm trong thành phần giao diện (hàm nhập) hay thành phần dữ liệu (hàm đọc dữ liệu) kiểm tra tính hợp lệ (hàm kiểm tra) và sau đó tiến hành xử lý (hàm xử lý) nếu cần thiết để cho ra kết quả mà sẽ được trình bày cho người dùng xem qua các hàm trong thành phần giao diện (hàm xuất) hoặc lưu trữ lại qua các hàm trong thành phần dữ liệu (hàm ghi).

***Bảng 1.2**: Danh sách các hàm cùng ý nghĩa tương ứng*

STT	Thành phần	Hàm	Ý nghĩa	Ghi chú
1	Thành phần giao diện	Hàm nhập Hàm xuất	Nhập yêu cầu, dữ liệu nguồn. Xuất kết quả đã xử lý	Cần xác định hình thức nhập/ xuất và tổ chức dữ liệu tương ứng
2	Thành phần xử lý	Hàm kiểm tra Hàm xử lý	Kiểm tra tính hợp lệ của dữ liệu. Xử lý tính toán, phát sinh, biến đổi trên dữ liệu	Sử dụng hàm nhập, hàm đọc. Sử dụng hàm nhập, hàm đọc, hàm xuất, hàm ghi
3	Thành phần dữ liệu	Hàm đọc Hàm ghi	Đọc dữ liệu từ bộ nhớ phụ vào bộ nhớ chính. Ghi dữ liệu từ bộ nhớ chính	Cần xác định cáchh thức tổ chức lưu trữ dữ liệu

Thế nào là PM tốt ?
=> PM tốt là phần mềm phải đáp ứng các chức năng theo yêu cầu, có hiệu năng tốt, có khả năng bảo trì, đáng tin cậy và được ng sử dụng chấp nhận

			vào bộ nhớ phụ	Tính chất cần có của phần mềm - Tính đúng đắn - Tính tiến hóa - Tính hiệu quả - TÍnh tiện dụng - Tính tương thích - TÍnh tái sử dụng

1.2. Chất lượng phần mềm

1.2.1. Tính đúng đắn

Tính đúng đắn của phần mềm được thể hiện ở chổ sản phẩm đó thực hiện đầy đủ và chính xác các yêu cầu của người dùng. Tính đúng đắn ở đây cần phải hiểu theo nghĩa rộng là chương trình cần phải thực hiện được trong cả những trường hợp mà dữ liệu đầu vào là không hợp lệ.

Ví dụ, nếu một trong số các chức năng của phần mềm là sắp xếp một tập tin có số lượng mẫu tin tùy ý theo một cột tùy ý theo chiều tăng hoặc giảm thì những trường hợp sau là vi phạm tính đúng đắn của chương trình:

- Không thể thực hiện được (treo máy) khi tập tin rỗng (không có mẫu tin nào).
- Không thể thực hiện hoặc thực hiện nhưng cho kết quả sai khi các mẫu tin có hơn 100 cột hoặc có quá nhiều mẫu tin.
- Không thể thực hiện hoặc cho kết quả sai khi các cột có chiều dài lớn hơn 125 bytes.
- Không thể sắp xếp theo chiều tăng dần….

Tính đúng đắn của một sản phẩm phần mềm được xác minh qua các căn cứ sau đây:

- Tính đúng đắn của thuật toán.
- Tính tương đương của chương trình với thuật toán. Thuật toán có thể đúng nhưng chương trình lập ra không tương đương với thuật toán nên khi thực hiện sẽ cho kết quả sai.
- Tính đúng đắn của chương trình có thể được chứng minh trực tiếp trong văn bản của chương trình.

- Tính đúng đắn cũng có thể được khẳng định dần qua việc kiểm thử, việc áp dụng chương trình trong một khoảng thời gian dài trên diện rộng và với tần suất sử dụng cao.

1.2.2. Tính tiến hóa

Cho phép người dùng có thể khai báo các thay đổi về qui định với phần mềm tùy theo các thay đổi trong thế giới thực liên quan (thay qui định về số sách mượn tối đa, công thức tính tiền phạt, công thức tính tiền điện…)

Sản phẩm có thể mở rộng, tăng cường về mặt chức năng một cách dễ dàng.

1.2.3. Tính hiệu quả

Tính hiệu quả của một sản phẩm phần mềm được xác định qua các tiêu chuẩn sau:

- Hiệu quả kinh tế hoặc ý nghĩa, giá trị thu được do áp dụng sản phẩm đó.
- Tốc độ xử lý của phần mềm (v) tính bằng tỉ lệ giữa khối lượng đối tượng cần phải xử lý (m) và tổng thời gian (t) cần thiết để xử lý các đối tượng đó.
- Sử dụng tối ưu tài nguyên của máy tính (CPU, bộ nhớ…)

1.2.4. Tính tiện dụng

Sản phẩm phải tính đến những yếu tố tâm lý sau đây của người dùng:

- Dễ học, có giao diện trực quan tự nhiên.
- Dễ thao tác,…

1.2.5. Tính tương thích

Trao đổi dữ liệu với các phần mềm khác có liên quan (nhận danh mục sách từ tập tin Excel, gửi báo cáo tổng kết năm học đến phần mềm WinFax, …)

- Giao tiếp nội bộ
- Giao tiếp bên ngoài

1.2.6. Tính tái sử dụng

Sản phẩm phần mềm có thể áp dụng cho nhiều lĩnh vực theo nhiều chế độ làm việc khác nhau.

- Các phần mềm cùng lớp
- Các phần mềm khác lớp

1.3. Công nghệ phần mềm

1.3.1. Sự ra đời

Vào những năm 1950 khi máy tính ra đời chính thức (không chỉ được dùng trong các phòng thí nghiệm mà bắt đầu ứng dụng trong hoạt động xã hội) các phần mềm đầu tiên cũng được ra đời với số lượng còn rất ít ỏi và chủ yếu phục vụ cho lĩnh vực tính toán (đặc biệt trong quốc phòng).

Đến những năm 1960, trãi qua 10 năm phát triển số lượng các phần mềm đã tăng lên rất nhiều và được ứng dụng rộng rãi trong nhiều lĩnh vực. Vào thời điểm này phát sinh một vấn đề mà các chuyên gia gọi là "cuộc khủng hoảng phần mềm". Cuộc khủng hoảng phần mềm thể hiện 2 yếu tố chính:

- Số lượng các phần mềm tăng vọt (do sự phát triển của phần cứng: tăng khả năng, giá thành hạ)
- Có quá nhiều khuyết điểm trong các phần mềm được dùng trong xã hội

- Thực hiện không đúng yêu cầu (tính toán sai, không ổn định…)
- Thời gian bảo trì, nâng cấp quá lâu, tốn chi phí cao, hiệu quả thấp.
- Khó sử dụng
- Thực hiện chậm
- Khó chuyển đổi dữ liệu giữa các phần mềm
- …..

Để giải quyết vấn đề trên một hội nghị đã được triệu tập đề bàn về cách giải quyết. Hội nghị đã tiến hành xem xét, phân tích và xác định nguyên nhân gây ra cuộc khủng hoảng phần mềm. Kết luận như sau:

- Việc tăng vọt của số lượng phần mềm là điều hợp lý và điều này sẽ còn tiếp diễn.
- Các khuyết điểm của phần mềm có nguồn gốc chính từ phương pháp, cách thức tiến hành xây dựng phần mềm:
    - Cảm tính: mỗi người theo một phương pháp riêng.
    - Thô sơ, đơn giản: chỉ tập trung vào việc lập trình mà ít quan tâm đến các công việc cần làm khác trước khi lập trình (khảo sát hiện trạng, phân tích yêu cầu, thiết kế…).
    - Thủ công: công cụ hỗ trợ chính khi xây dựng phần mềm chỉ là trình biên dịch.

Với các kết luận như trên, hội nghị đã đề xuất khai sinh một ngành khoa học mới: Công nghệ phần mềm với nhiệm vụ chính là nghiên cứu về các phương pháp tiến hành xây dựng phần mềm.

1.3.2. Định nghĩa

Công nghệ phần mềm là một lĩnh vực nghiên cứu của tin học nhằm đề xuất các nguyên lý, phương pháp, công cụ, cách

tiếp cận phục vụ cho việc thiết kế, cài đặt các sản phẩm phần mềm đạt được đầy đủ các yêu cầu về chất lượng phần mềm.

Do quá trình tiến hóa của ngành công nghệ phần mềm nên khái niệm về nó cũng thay đổi theo thời gian. Hơn nữa do đây là một lĩnh vực mới nên các khái niệm vẫn còn phụ thuộc rất nhiều vào quan điểm chủ quan của từng người khác nhau. Cụ thể như sau:

- Bauer[1969]: việc thiết lập và sử dụng các nguyên lý công nghệ đúng đắn để thu được phần mềm một cách kinh tế vừa tin cậy vừa làm việc hiệu quả trên các máy thực.
- Ghezzi[1991]: là một lĩnh vực của khoa học máy tính liên quan đến việc xây dựng các phần mềm vừa lớn vừa phức tạp bởi một hay một số nhóm kỹ sư.
- IEEE[1993]:
    1. Việc áp dụng phương pháp tiếp cận có hệ thống, bài bản và được lượng hóa trong phát triển, vận hành và bảo trì phần mềm.
    2. Nghiên cứu các phương pháp tiếp cận được dùng trong (1).
- Sommervile[1995]: là lĩnh vực liên quan đến lý thuyết, phương pháp và công cụ dùng cho phát triển phần mềm.
- Kawamura[1995]: là lĩnh vực học vấn về các kỹ thuật, phương pháp luận công nghệ học (lý luận và kỹ thuật được hiện thực hóa trên các nguyên lý, nguyên tắc xác định) trong toàn bộ quy trình phát triển phần mềm nhằm nâng cao cả chất và lượng của sản xuất phần mềm.

- Pressman[1995]: là bộ môn tích hợp cả qui trình, các phương pháp, các công cụ để phát triển phần mềm máy tính.

Có thể định nghĩa tóm tắt về công nghệ phần mềm như sau: Công nghệ phần mềm là một nghành khoa học nghiên cứu về việc xây dựng các phần mềm có chất lượng trong khoảng thời gian và chi phí hợp lý.

Mục tiêu nghiên cứu được chia thành 2 phần rõ nét:

1. Xây dựng phần mềm có chất lượng.
2. Xây dựng phần mềm trong thời gian và chi phí hợp lý.

1.3.3. Đối tượng nghiên cứu

Hướng đến việc xây dựng các phần mềm có chất lượng như đã nêu, ngành công nghệ phần mềm đưa ra 3 đối tượng nghiên cứu chính: Qui trình công nghệ, Phương pháp phát triển, Công cụ và môi trường phát triển phần mềm.

- Qui trình công nghệ phần mềm: Hệ thống các giai đoạn mà quá trình phát triển phần mềm phải trãi qua. Với mỗi giai đoạn cần xác định rõ mục tiêu, kết quả nhận từ giai đoạn trước đó cũng chính là kết quả chuyển giao cho giai đoạn kết tiếp.
- Phương pháp phát triển phần mềm: Hệ thống các hướng dẫn cho phép từng bước thực hiện một giai đoạn nào đó trong qui trình công nghệ phần mềm.
- Công cụ và môi trường phát triển phần mềm: Hệ thống các phần mềm trợ giúp chính trong lĩnh vực xây dựng phần mềm. Các phần mềm này sẽ hỗ trợ các chuyên viên tin học trong các bước xây dựng phần mềm theo một phương pháp nào đó với một qui trình được chọn trước.

2. QUI TRÌNH CÔNG NGHỆ PHẦN MỀM

Như đã nói để xây dựng được phần mềm có chất lượng quá trình phát triển phải trãi qua rất nhiều giai đoạn. Mỗi giai đoạn có mục tiêu và kết quả chuyển giao xác định. Trình tự thực hiện các giai đoạn này chính là chu kỳ sống của một phần mềm.

Nói cách khác, chu kỳ sống của một phần mềm là khoảng thời gian mà trong đó một sản phẩm phần mềm được phát triển, sử dụng và mở rộng cho đến khi sản phẩm phần mềm đó không còn được sử dụng nữa.

Chu kỳ sống của phần mềm được phân chia được phân chia thành các pha chính như: xác định, phát triển, kiểm thử, bảo trì (vận hành). Phạm vi và thứ tự các pha khác nhau tùy theo từng mô hình cụ thể.

2.1. Các bước cơ bản trong xây dựng phần mềm

2.1.1. Xác định

(hay còn gọi là dự án)

Đây là bước hình thành bài toán hoặc đề tài. Ở bước này thiết kế trưởng hoặc phân tích viên hệ thống phải biết được vai trò của phần mềm cần phát triển trong hệ thống, đồng thời phải ước lượng công việc, lập lịch biểu và phân công công việc.

(Xác định được yêu cầu của khách hàng)

Bên cạnh đó chúng ta phải biết người đặt hàng muốn gì. Các yêu cầu cần phải được thu thập đầy đủ và được phân tích theo chiều ngang (rộng) và chiều dọc (sâu). Công cụ sử dụng chủ yếu ở giai đoạn này là các lược đồ, sơ đồ phản ánh rõ các thành phần của hệ thống và mối liên quan giữa chúng với nhau.

2.1.2. Phát triển

Dựa vào các nội dung đã xác định được, nhóm phát triển phần mềm dùng ngôn ngữ đặc tả hình thức (dựa trên các kiến trúc toán học) hoặc phi hình thức (tựa ngôn ngữ tự nhiên) hoặc kết hợp cả hai để mô tả những yếu tố sau đây của chương trình:

- Giá trị nhập, giá trị xuất.
- Các phép biến đổi
- Các yêu cầu cần đạt được ở mỗi điểm của chương trình.

Phần đặc tả chỉ quan tâm chủ yếu đến giá trị vào, ra chứ không quan tâm đến cấu trúc và nội dung các thao tác cần thực hiện.

Sau bước thiết kế là bước triển khai các đặc tả chương trình thành một sản phẩm phần mềm dựa trên một ngôn ngữ lập trình cụ thể. Trong giai đoạn này các lập trình viên sẽ tiến hành cài đặt các thao tác cần thiết để thực hiện đúng các yêu cầu đã được đặc tả.

Công việc cuối cùng của giai đoạn phát triển là chúng ta cần phải chứng minh tính đúng đắn của chương trình sau khi đã tiến hành cài đặt. Tuy nhiên thông thường ở bước này chúng ta coi các chương trình như những hộp đen. Vấn đề đặt ra là xây dựng một cách có chủ đích các tập dữ liệu nhập khác nhau để giao cho chương trình thực hiện rồi dựa vào kết quả thu được để đánh giá chương trình. Công việc như trên được gọi là kiểm thử chương trình.

Công việc kiểm thử nhằm vào các mục tiêu sau:

- Kiểm tra để phát hiện lỗi của chương trình. Lưu ý rằng kiểm thử không đảm bảo tuyệt đối tính đúng

đắn của chương trình do bản chất quy nạp không hoàn toàn của cách làm.

- Kiểm tra tính ổn định, hiệu quả cũng như khả năng tối đa của chương trình.

Tùy theo mục đích mà người ta thiết kế các tập dữ liệu thử sao cho có thể phủ hết các trường hợp cần quan tâm.

2.1.3. Bảo trì (Vận hành)

Công việc quản lý việc triển khai và sử dụng phần mềm cũng là một vấn đề cần được quan tâm trong qui trình phát triển phần mềm. Trong quá trình xây dựng phần mềm, toàn bộ các kết quả phân tích, thiết kế, cài đặt và hồ sơ liên quan cần phải được lưu trữ và quản lý cẩn thận nhằm đảm bảo cho công việc được tiến hành một cách hiệu quả nhất và phục vụ cho công việc bảo trì phần mềm về sau.

Như vậy công việc quản lý không chỉ dừng lại trong quá trình xây dựng phần mềm mà trái lại còn phải được tiến hành liên tục trong suốt quá trình sống của nó.

2.2. Một số mô hình triển khai xây dựng phần mềm

Có nhiều mô hình cận khác nhau để triển khai các bước cơ bản trong quá trình phát triển phần mềm. Mỗi mô hình sẽ chia vòng đời của phần mềm theo một cách khác nhau nhằm đảm bảo qui trình phát triển phần mềm sẽ dẫn đến thành công. Trong phần tiếp theo của giáo trình chúng ta sẽ tìm hiểu qua các mô hình phát triển phần mềm tiêu biểu nhất đang được áp dụng.

2.2.1. Mô hình thác nước:

Mô hình thác nước là một trong những mô hình đầu tiên và phổ biến được áp dụng trong quá trình phát triển phần mềm. Mô hình này chia quá trình phát triển phần mềm thành

những giai đoạn tuần tự nối tiếp nhau. Mỗi giai đoạn sẽ có một mục đích nhất định. Kết quả cuả giai đoạn trước sẽ là thông tin đầu vào cho giai đoạn tiếp theo sau. Tùy theo qui mô của phần mềm cần phát triển mà mô hình thác nước sẽ có những biến thể khác nhau như sau:

- ❖ Qui trình 2 giai đoạn: Là qui trình đơn giản nhất. Theo qui trình này việc phát triển phần mềm chỉ trãi qua 2 giai đoạn:
  - o Xác định yêu cầu: Được tiến hành ngay khi có nhu cầu về việc xây dựng phần mềm.
    - Mục tiêu: Xác định chính xác các yêu cầu đặt ra cho phần mềm sẽ xây dựng.
    - Kết quả nhận: Thông tin về hoạt động của thế giới thực.
    - Kết quả chuyển giao: Danh sách các yêu cầu (công việc sẽ thực hiện trên máy tính) cùng với các thông tin miêu tả chi tiết về các yêu cầu (cách thức thực hiện trong thế giới thực).
  - o Lập trình (cài đặt): Được tiến hành ngay sau khi kết thúc việc xác định yêu cầu.
    - Mục tiêu: Tạo lập phần mềm mong muốn theo yêu cầu.
    - Kết quả nhận: Danh sách các yêu cầu cùng các thông tin có liên quan.
    - Kết quả chuyển giao: Chương trình nguồn của phần mềm với cấu trúc cơ sở dữ liệu tương ứng (nếu cần thiết) và chương trình thực hiện được trên máy tính (chương trình nguồn đã được biên dịch)
- ❖ Qui trình 3 giai đoạn: Là qui trình cải tiến của qui trình 2 giai đoạn bằng cách bổ sung thêm một giai đoạn

trung gian mới giữa xác định yêu cầu và lập trình (có sửa đổi)

- Xác định yêu cầu: được tiến hành ngay khi có nhu cầu về việc xây dựng phần mềm.
  - Mục tiêu: Xác định chính xác các yêu cầu đặt ra cho phần mềm sẽ xây dựng.
  - Kết quả nhận: Thông tin về hoạt động của thế giới thực.
  - Kết quả chuyển giao: Danh sách các yêu cầu (công việc sẽ thực hiện trên máy tính) cùng với các thông tin miêu tả chi tiết về các yêu cầu (cách thức thực hiện trong thế giới thực)
- Thiết kế: Được tiến hành ngay sau khi kết thúc việc xác định yêu cầu.
  - Mục tiêu: Mô tả các thành phần của phần mềm (mô hình của phần mềm) trước khi tiến hành cài đặt.
  - Kết quả nhận: Danh sách các yêu cầu và thông tin liên quan.
  - Kết quả chuyển giao:
    - Mô tả thành phần giao diện: các hàm nhập/xuất, cấu trúc dữ liệu nhập/xuất.
    - Mô tả thành phần xử lý: các hàm kiểm tra xử lý.
    - Mô tả thành phần dữ liệu: các hàm đọc/ ghi, tổ chức lưu trữ trên bộ nhớ phụ.
- Lập trình (cài đặt): Được tiến hành ngay sau khi kết thúc việc thiết kế.
  - Mục tiêu: Tạo lập phần mềm theo yêu cầu.
  - Kết quả nhận: Mô hình phần mềm

- Kết quả chuyển giao: Chương trình nguồn của phần mềm với cấu trúc cơ sở dữ liệu tương ứng (nếu cần thiết) và chương trình thực hiện được trên máy tính (chương trình nguồn đã được biên dịch)

❖ Qui trình 4 giai đoạn: Là qui trình cải tiến của qui trình phía trước bằng cách bổ sung thêm một giai đoạn mới giữa xác định yêu cầu và thiết kế (có sửa đổi)

o Xác định yêu cầu: Được tiến hành ngay khi có nhu cầu về việc xây dựng phần mềm.

- Mục tiêu: Xác định chính xác các yêu cầu đặt ra cho phần mềm sẽ xây dựng.
- Kết quả nhận: Thông tin về hoạt động của thế giới thực.
- Kết quả chuyển giao: Danh sách các yêu cầu (công việc sẽ thực hiện trên máy tính) cùng với các thông tin miêu tả chi tiết về các yêu cầu (cách thức thực hiện trong thế giới thực)

o Phân tích: được tiến hành ngay sau khi kết thúc việc xác định yêu cầu.

- Mục tiêu: Mô tả lại thế giới thực thông qua các mô hình (mô hình thế giới thực) trước khi thiết kế.
- Kết quả nhận: Danh sách các yêu cầu cùng các thông tin có liên quan.
- Kết quả chuyển giao:
  - Mô hình xử lý (hệ thống các công việc trong thế giới thực cùng với quan hệ giữa chúng)

        - Mô hình dữ liệu (hệ thống các loại thông tin được sử dụng trong thế giới thực cùng với quan hệ giữa chúng)
        - Các mô hình khác (không gian, thời gian, con người…) nếu cần thiết.

    - Thiết kế: Được tiến hành ngay sau khi kết thúc việc phân tích.

        - Mục tiêu: Mô tả các thành phần của phần mềm (mô hình của phần mềm) trước khi tiến hành cài đặt.
        - Kết quả nhận: Mô hình thế giới thực.
        - Kết quả chuyển giao:

            - Mô tả thành phần giao diện: các hàm nhập/xuất, cấu trúc dữ liệu nhập/xuất.
            - Mô tả thành phần xử lý: các hàm kiểm tra xử lý.
            - Mô tả thành phần dữ liệu: các hàm đọc/ghi, tổ chức lưu trữ trên bộ nhớ phụ.

    - Lập trình (cài đặt): Được tiến hành ngay sau khi kết thúc việc thiết kế.

        - Mục tiêu: Tạo lập phần mềm theo yêu cầu
        - Kết quả nhận: Mô hình phần mềm
        - Kết quả chuyển giao: Chương trình nguồn của phần mềm với cấu trúc cơ sở dữ liệu tương ứng (nếu cần thiết) và chương trình thực hiện được trên máy tính (chương trình nguồn đã được biên dịch)

- Qui trình 5 giai đoạn: Là qui trình cải tiến của qui trình phía trước bằng cách bổ sung thêm một giai đoạn mới

sau giai đoạn lập trình nhằm tăng cường độ tin cậy của phần mềm.

- Xác định yêu cầu: Được tiến hành ngay khi có nhu cầu về việc xây dựng phần mềm.
  - Mục tiêu: Xác định chính xác các yêu cầu đặt ra cho phần mềm sẽ xây dựng.
  - Kết quả nhận: Thông tin về hoạt động của thế giới thực.
  - Kết quả chuyển giao: Danh sách các yêu cầu (công việc sẽ thực hiện trên máy tính) cùng với các thông tin miêu tả chi tiết về các yêu cầu (cách thức thực hiện trong thế giới thực)
- Phân tích: được tiến hành ngay sau khi kết thúc việc xác định yêu cầu.
  - Mục tiêu: Mô tả lại thế giới thực thông qua các mô hình (mô hình thế giới thực) trước khi thiết kế.
  - Kết quả nhận: Danh sách các yêu cầu cùng các thông tin có liên quan.
  - Kết quả chuyển giao:
    - Mô hình xử lý (hệ thống các công việc trong thế giới thực cùng với quan hệ giữa chúng)
    - Mô hình dữ liệu (hệ thống các loại thông tin được sử dụng trong thế giới thực cùng với quan hệ giữa chúng)
    - Các mô hình khác (không gian, thời gian, con người…) nếu cần thiết.
- Thiết kế: Được tiến hành ngay sau khi kết thúc việc phân tích.

- Mục tiêu: Mô tả các thành phần của phần mềm (mô hình của phần mềm) trước khi tiến hành cài đặt.
- Kết quả nhận: Mô hình thế giới thực.
- Kết quả chuyển giao:
  - Mô tả thành phần giao diện: các hàm nhập/xuất, cấu trúc dữ liệu nhập/xuất.
  - Mô tả thành phần xử lý: các hàm kiểm tra xử lý.
  - Mô tả thành phần dữ liệu: các hàm đọc/ ghi, tổ chức lưu trữ trên bộ nhớ phụ.

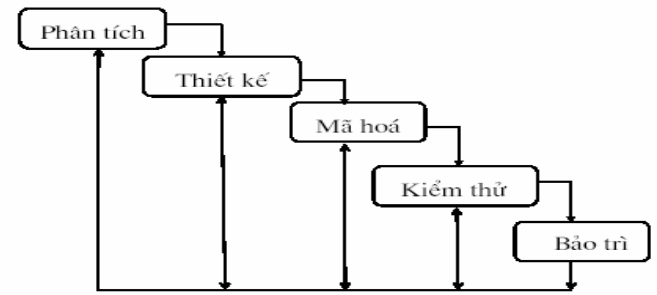
o Lập trình (cài đặt): Được tiến hành ngay sau khi kết thúc việc thiết kế.

- Mục tiêu: Tạo lập phần mềm theo yêu cầu.
- Kết quả nhận: Mô hình phần mềm.
- Kết quả chuyển giao: Chương trình nguồn của phần mềm với cấu trúc cơ sở dữ liệu tương ứng (nếu cần thiết) và chương trình thực hiện được trên máy tính (chương trình nguồn đã được biên dịch).

o Kiểm thử: Được tiến hành ngay sau khi đã có kết quả (từng phần) của việc lập trình.
- Mục tiêu: Tăng độ tin cậy của phần mềm.
- Kết quả nhận:
  - Danh sách yêu cầu.
  - Mô hình phần mềm.
  - Phần mềm.
- Kết quả chuyển giao: Phần mềm với độ tin cậy cao (đã tìm và sửa lỗi).

o Bảo trì: Công việc của giai đoạn bao gồm việc cài đặt và vận hành phần mềm trong thực tế.

- Mục tiêu: đảm bảo phần mềm vận hành tốt
- Kết quả nhận: phần mềm đã hoàn thàng
- Kết quả chuyển giao: các phản ánh của khách hàng trong quá trình sử dụng phần mềm.



Mô hình vòng đời cổ điển (thác nước)

Nhận xét:

Mô hình thác nước giúp chúng ta có thể dễ dàng phân chia quá trình xây dựng phần mềm thành những giai đoạn hoàn toàn độc lập nhau. Tuy nhiên, các dự án lớn hiếm khi tuân theo dòng chảy tuần tự của mô hình vì thường phải lặp lại các bước để nâng cao chất lượng. Hơn nữa, khách hàng hiếm khi tuyên bố hết các yêu cầu trong giai đoạn phân tích.

Mô hình này cũng có một hạn chế là chúng ta rất khó thực hiện các thay đổi một khi đã thực hiện xong một giai đoạn nào đó. Điều này làm cho việc xây dựng phần mềm rất khó thay đổi các yêu cầu theo ý muốn của khách hàng. Do đó, phương pháp này chỉ thích hợp cho những trường hợp mà chúng ta đã hiểu rất rõ các yêu cầu của khách hàng.

Chú ý: Mô hình thác nước có thể được cải tiến bằng cách cho phép quay lui khi phát hiện lỗi trong giai đoạn phía trước.

2.2.2. Mô hình bản mẫu phần mềm

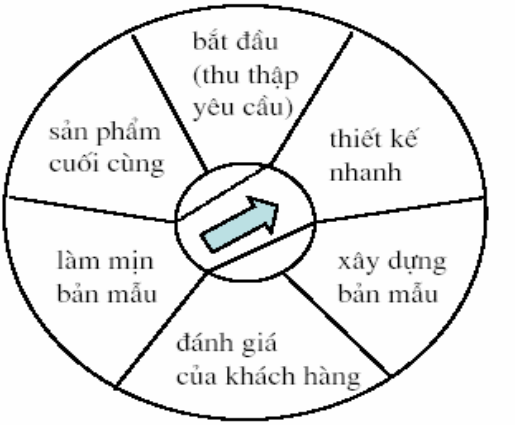
Tương tự như mô hình thác nước với bổ sung vào các giai đoạn thực hiện phần mềm mẫu ngay khi xác định yêu cầu nhằm mục tiêu phát hiện nhanh các sai sót về yêu cầu. Các giai đoạn trong mô hình bản mẫu phần mềm có thể tiến hành lặp đi lặp lại chứ không nhất thiết phải theo trình tự nhất định.

Ngay sau khi giai đoạn xác định yêu cầu, nhà phát triển phần mềm đưa ra ngay một bản thiết kế sơ bộ và tiến hành cài đặt bản mẫu đầu tiên và chuyển cho người sử dụng. Bản mẫu này chỉ nhằm để miêu tả cách thức phần mềm hoạt động cũng như cách người sử dụng tương tác với hệ thống.

Người sử dụng sau khi xem xét sẽ phản hồi thông tin cần thiết lại cho nhà phát triển. Nếu người sử dụng đồng ý với bản mẫu đã đưa thì người phát triển sẽ tiến hành cài đặt thực sự. Ngược lại cả hai phải quay lại giai đoạn xác định yêu cầu. Công việc này được lặp lại liên tục cho đến khi người sử dụng đồng ý với các bản mẫu do nhà phát triển đưa ra.

Như vậy đây là một hướng tiếp cận tốt khi các yêu cầu chưa rõ ràng và khó đánh giá được tính hiệu quả của các thuật toán. Tuy nhiên, mô hình này cũng có nhược điểm là tính cấu trúc không cao do đó khách hàng dễ mất tin tưởng.

Chỉ nên áp dụng với những hệ thống có tương tác ở mức độ vừa và nhỏ; một phần của hệ thống lớn hoặc có thời gian chu kỳ tồn tại ngắn

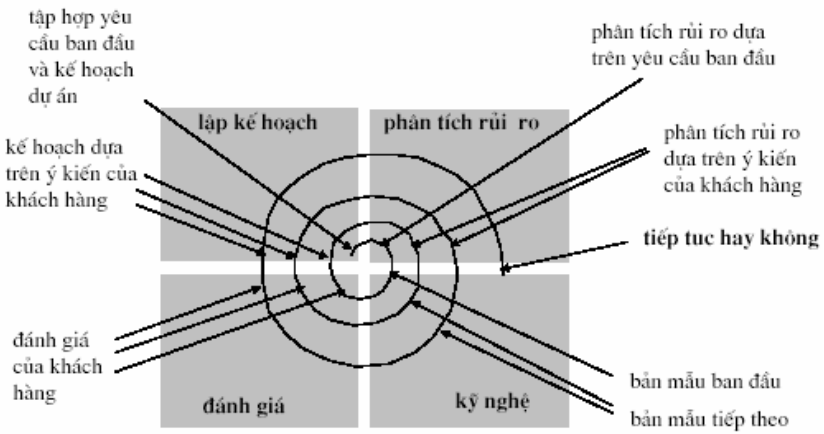


Mô hình làm bản mẫu

2.2.3. Mô hình xoắn ốc

Mô hình này chính là sự kết hợp của mô hình bản mẫu thiết kế và mô hình thác nước được lặp lại nhiều lần. Ở lần lặp tiếp theo hệ thống sẽ được tìm hiểu và xây dựng hoàn thiện hơn ở lần lặp trước đó.

Ở mỗi lần lặp các yêu cầu của người sử dụng sẽ được hiểu ngày càng rõ ràng hơn và các bản mẫu phần mềm cũng ngày một hoàn thiện hơn. Ngoài ra ở cuối mỗi lần lặp sẽ có thêm công đoạn phân tích mức độ rủi ro để quyết định xem có nên đi tiếp theo hướng này nữa hay không.



Mô hình xoắn ốc

Mô hình này phù hợpvới các hệ thống phần mềm lớn do có khả năng kiểm soát rủi ro ở từng bước tiến hóa. Tuy nhiên vẫn chưa được sử dụng rộng rãi như mô hình thác nước hoặc bản mẫu do đòi hỏi năng lực quản lý, năng lực phân tích rủi ro cao.

3. CÁC PHƯƠNG PHÁP XÂY DỰNG PHẦN MỀM

3.1. Tổng quan

3.1.1. Khái niệm

Để tiến hành xây dựng một phần mềm, chúng ta có thể áp dụng nhiều phương pháp khác nhau. Mỗi phương pháp có những ưu và khuyết điểm riêng và phù hợp với từng loại phần mềm cụ thể.

Mỗi phương pháp sẽ có những hướng dẫn cụ thể các công việc cần phải thực hiện trong từng giai đoạn trong quy trình xây dựng phần mềm.

Bên cạnh đó mỗi phương pháp cũng sẽ quy định những cách thức khác nhau để trình bày các kết quả thu được trong quá trình xây dựng phần mềm. Những quy định này có tính chất như là ngôn ngữ thống nhất để các thành viên tham gia xây dựng phần mềm có thể trao đổi thông tin trong việc xây dựng phần mềm.

3.1.2. Phân loại

Các phương pháp xây dựng phần mềm được chia làm 02 nhóm khác nhau dựa vào tính chất của công việc cần thực hiện.

- ❖ Phương pháp xây dựng:
  - Phương pháp hướng chức năng
  - Phương pháp hướng dữ liệu
  - Phương pháp hướng đối tượng
- ❖ Phương pháp tổ chức quản lý
  - Xây dựng phương án
  - Tổ chức nhân sự
  - Ước lượng rủi ro, chi phí
  - Lập và theo dõi kế hoạch triển khai.

Trong phần tiếp theo của giáo trình này, chúng ta chỉ quan tâm đến các phương pháp xây dựng. Về phương pháp tổ chức quản lý chúng ta có thể tham khảo trong giáo trình "Quản lý dự án xây dựng các hệ thống thông tin".

3.2. Các phương pháp xây dựng phần mềm

3.2.1. Cách tiếp cận

a) Từ trên xuống

Đây là cách giải quyết vấn đề theo hướng phân tích. Khi tiến hành xây dựng phần mềm theo cách này, chúng ta bắt đầu với những thành phần chính của hệ thống. Sau đó, các thành phần này sẽ được phân tích thành các thành phần chi tiết và cụ thể hơn. Quá trình phân tích này sẽ kết thúc khi các kết quả thu được có mức độ phức đạp đúng với ý muốn của nhà xây dựng phần mềm.

b) Từ dưới lên

Ngược lại với phương pháp từ trên xuống, phương pháp từ dưới lên là cách giải quyết vấn đề theo hướng tổng hợp. Với phương pháp này, chúng ta tiến hành xây dựng những thành phần chi tiết, cụ thể mà mà chúng ta dự tính là sẽ có trong hệ thống. Sau đó, các nhà phát triển phần mềm sẽ tiến hành kết hợp các thành phần chi tiết này lại với nhau để tạo nên các thành phần chính mà hệ thống cần phải có.

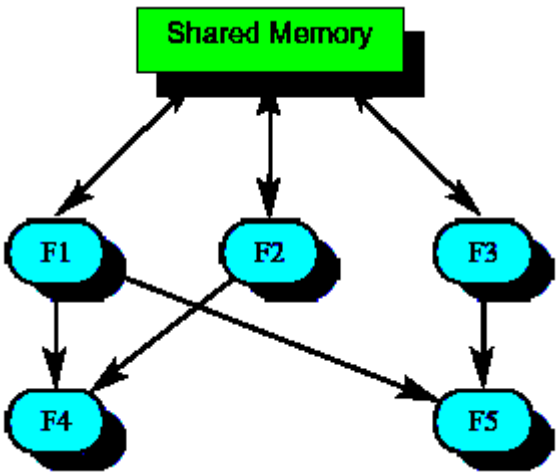
3.2.2. Cách tiến hành

a) Phương pháp hướng chức năng

Với phương pháp này công việc xây dựng phần mềm được thực hiện dựa trên các chức năng mà hệ thống cần thực hiện. Hay nói cách khác chúng ta chú trọng đến thành phần xử lý của hệ thống:

- Các thao tác tính toán
- Các thao tác phát sinh
- Các thao tác biến đổi….

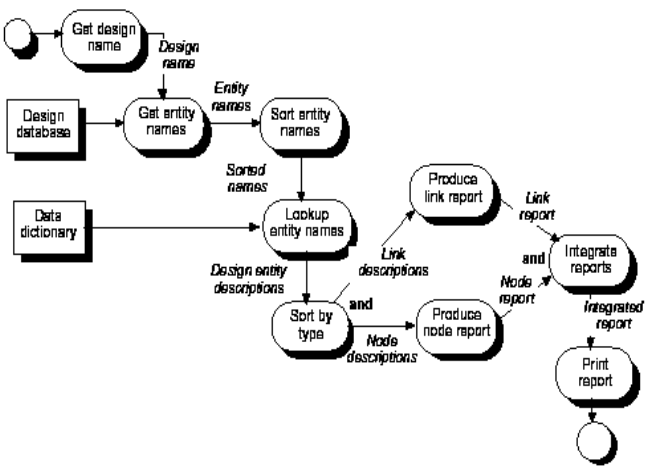
Phương pháp chung để giải quyết vấn đề là áp dụng nguyên lý "chia để trị". Khi tiến hành xây dựng phần mềm theo phương pháp này, chúng ta sẽ chia các công việc lớn mà hệ thống cần thực hiện hành các công việc nhỏ hơn độc lập nhau. Việc phân chia các công việc được tiến hành cho đến khi các công việc thu được đủ nhỏ để chúng ta có thể tiến hành xây dựng hoàn chỉnh. Hình dưới: Minh họa cách tiếp cận theo hướng chức năng.



Phương pháp hướng chức năng chú trọng đến cách để giải quyết vấn đề nhưng không có khả năng che dấu các thông tin trạng thái của hệ thống. Điều này dẫn đến việc các chức năng trong hệ thống không tương thích với nhau trong việc thực hiện thay đổi các thông tin trong hệ thống. Chính vì vậy mà cách tiếp cận này chỉ thích hợp khi trong hệ thống có rất ít thông tin cần phải quản lý và chia sẻ giữa các chức năng với nhau. Để mô hình hóa cách xử lý thông tin trong hệ thống dùng lược đồ dòng dữ liệu (Data Flow Diagrams).

DFD là một công cụ đơn giản và hữu ích để miêu tả cách thức hoạt động của hệ thống. DFD sử dụng các ký hiệu sau để mô tả hệ thống:

- Ô vuông có góc tròn được dùng để biểu diễn các chức năng của hệ thống.
- Ô vuông dùng để biểu diễn thành phần dữ liệu trong hệ thống.
- Hình tròn dùng để biểu diễn các thành phần bên ngoài có giao tiếp với hệ thống.
- Dấu mũi tên dùng để biểu diễn hướng di chuyển của dữ liệu.
- Các từ khóa "and" và "or" dùng để liên kết các dòng dữ liệu khi cần thiết.



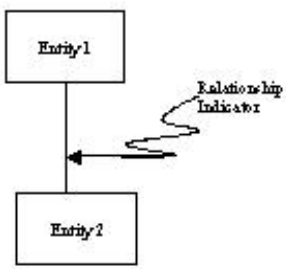
b) Phương pháp hướng dữ liệu

Ngược lại với phương pháp hướng chức năng, phương pháp hướng dữ liệu chú trọng nhiều đến thành phần dữ liệu cần phải xử lý trong hệ thống:

- Tổ chức dữ liệu
- Khối lượng lưu trữ
- Tốc độ truy xuất
- …

Khi tiến hành thiết kế theo phương pháp hướng dữ liệu chúng ta bắt đầu với việc thiết kế các cấu trúc dữ liệu cần thiết có trong bài toán, sau đó mới tiến hành thiết kết các thao tác để vận hành trên các cấu trúc dữ liệu đã thiết kế.

Phương pháp này đặc biệt chỉ thích hợp trong các loại phần mềm chỉ có chức năng chính là lưu trữ và thao tác trên các loại dữ liệu. Hạn chế của nó là không quan tâm đến các



chức năng mà hệ thống cần phải đáp ứng. Điều này dẫn đến việc có khả năng hệ thống sau khi thiết kế không có đầy đủ các chức năng cần thiết.

Kết quả thu được sau khi thiết kế theo phương pháp hướng dữ liệu là mô hình thực thể kết hợp (Entity Relationship Diagram). Một mô hình thực thể kết hợp điển hình gồm có 2 thành phần cơ bản: các thực thể và các mối kết hợp.

- Một thực thể là một đối tượng trong thế giới thực mà hệ thống có quan hệ, hoặc tương tác qua lại. Các thực thể được biểu diễn trong sơ đồ bằng các

hình vuông cùng với tên và có thể có thêm các thuộc tính của thực thể.

- Mối kết hợp biểu diễn sự kết hợp giữa hai hay nhiều thực thể. Mỗi mối kết hợp gồm có ba thành phần cơ bản:

    - Mối kết hợp giữa các thực thể được biểu diễn băng một đường thẳng nối giữa hai thực thể.
    - Tên của môi liên hệ dùng để miêu tả ý nghĩa của mối liên hệ.
    - Bản số ở hai đầu của mối kết hợp dùng để xác định con số tối đa và tối thiểu các thực thể liên quan đến mối kết hợp.

c) Phương pháp hướng đối tượng

Phương pháp thiết kế hướng đối tượng là sự kết hợp của phương pháp hướng dữ liệu và phương pháp hướng chức năng. Phương pháp này chú trọng đến cả thành phần dữ liệu và chức năng của hệ thống.

Theo phương pháp hướng đối tượng thì một hệ thống phần mềm là một tập hợp các đối tượng có khả năng tương tác với nhau. Các đối tượng chính là các sự vật và hiện tượng vật lý cũng như trừu tượng mà chúng ta có trong thế giới thực. Mỗi đối tượng có dữ liệu riêng được che dấu với thế giới bên ngoài và các thao tác mà đối tượng có thể thực hiện trên các thành phần dữ liệu của đối tượng.

Các đối tượng liên lạc, trao đổi thông tin với nhau bằng cách gửi các thông điệp cho nhau. Các thông điệp mà mỗi đối tượng có thể xử lý được gọi là giao diện của đối tượng. Khi đó mọi thao tác liên quan đến các đối tượng được phải thực hiện thông qua giao diện của đối tượng. Điều này giúp chúng ta

đảm bảo rằng các thông tin bên trong các đối tượng được bảo vệ một cách chắc chắn.

Chúng ta có thể sử dụng nhiều hệ thống ký hiệu khác nhau để mô tả các đối tượng của hệ thống cũng như mối liên hệ giữa chúng. Một trong số các hệ thống ký hiệu phổ biến hiện nay là hệ thốnng ký hiệu UML.

4. CÔNG CỤ VÀ MÔI TRƯỜNG PHÁT TRIỂN PHẦN MỀM

4.1. Mở đầu

4.1.1. Khái niệm

Các công cụ và môi trường phát triển phần mềm là các phần mềm hỗ trợ chính người phát triển trong quá trình xây dựng phần mềm. Các phần mềm này có tên gọi chung là CASE (Computer Aided Software Engineering) tools.

Trong quá trình phát triển phần mềm theo các quy trình trên, các CASE tools có thể hỗ trợ cụ thể cho một giai đoạn nào đó hay cũng có thể hỗ trợ một số giai đoạn, trong trường hợp này tên gọi chung thường là môi trường phát triển phần mềm-SDE (Software Development Environment).

Việc hỗ trợ của các CASE tools trong một giai đoạn bao gồm 2 hình thức chính:

- Cho phép lưu trữ, cập nhật trên kết quả chuyển giao với một phương pháp nào đó.
- Cho phép phát sinh ra kết quả chuyển giao cho giao đoạn kế tiếp.

4.2. Phần mềm hỗ trợ thực hiện các giai đoạn

4.2.1. Phần mềm hỗ trợ phân tích

- Công việc hỗ trợ chính

    - Soạn thảo các mô hình thế giới thực
    - Ánh xạ vào mô hình logic
- Các phần mềm: WinA&D, Analyst Pro,…

4.2.2. Phần mềm hỗ trợ thiết kế

- Công việc hỗ trợ chính
    - Soạn thảo các mô hình logic
    - Ánh xạ vào mô hình vật lý
- Các phần mềm: QuickUML, Power Designer, Oracle Designer…

4.2.3. Phần mềm hỗ trợ lập trình

- Công việc hỗ trợ chính
    - Quản lý các phiên bản (Dữ liệu, chương trình nguồn, giao diện)
    - Biên dịch
- Các phần mềm: Visual Studio, Visual Basic, Visual C++

4.2.4. Phần mềm hỗ trợ kiểm chứng

- Công việc hỗ trợ chính
    - Phát sinh tự động các bộ dữ liệu thử nghiệm
    - Phát hiện lỗi
- Các phần mềm: WinRuner

4.3. Phần mềm hỗ trợ tổ chức, quản lý việc triển khai

4.3.1. Xây dựng phương án

- Công việc hỗ trợ chính
    - Tạo lập phương án
    - Dự đoán rủi ro
    - Tính chi phí
- Các phần mềm: MS Project, Visio

4.3.2. Lập kế hoạch

- Công việc hỗ trợ chính
    - Xác định các công việc

    - Phân công
    - Lập lịch biểu
    - Theo dõi thực hiện
- Các phần mềm: MS Project, Visio

Chương 2: PHÂN TÍCH VÀ ĐẶC TẢ YÊU CẦU

1. Tổng quan

Phân tích yêu cầu là khâu kỹ thuật đầu tiên gồm nhiều bước nhỏ: *nghiên cứu khả thi, phân tích mô hình hóa, đặc tả thẩm định yêu cầu.* Gia đoạn này được tiến hành phối hợp giữa bên phát triển và khách hàng và nó có vai trò đặc biệt quan trọng trong tiến trình phát triển phần mềm.

Đây là bước hình thành bài toán hoặc đề tài. Ở bước này trưởng nhóm thiết kế và người phân tích hệ thống phải biết được người đặt hàng muốn gì. Các yêu cầu phải được thu thập đầy đủ và được phân tích theo chiều ngang (rộng) và dọc (sâu). Công cụ sử dụng chủ yếu ở giai đoạn này là các lược đồ, sơ đồ phản ánh rõ các đối tượng của hệ thống: lưu đồ (Flowchart), sơ đồ dòng dữ liệu (Data Flow diagram DFD), mạng thực thể-quan hệ (Entity-Relationship Network), sơ đồ cấu trúc phân cấp (Structural hierarchical schemes), mạng ngữ nghĩa (Semantic Network)

1.1 Quá trình phân tích

1.1.1 Phân tích phạm vi dự án

Người phân tích hệ thống dùng thuật ngữ phạm vi để chỉ trách nhiệm dự án phải thực thi. Ngược lại, phạm vi dự án là nhiệm vụ lớn và phức tạp được thực hiện bởi chương trình.

Để xác định phạm vi dự án, bằng xác định quá trình nghiệp vụ ứng dụng sẽ đối đầu. Đó là những phạm vi vấn đề của ứng dụng. Nói chung, có hai phần đối với bất kỳ giải pháp

nghiệp vụ: phần triển khai ứng dụng và phần thực hiện bởi con người hay chương trình. Định ra ranh giới ứng dụng tức là xác định qui trình trách nhiệm.

Một khi đã định nghĩa trách nhiệm của dự án:

- Chia trách nhiệm thành những nhiệm vụ con để đưa ra ý tưởng cho chính mình về bao nhiêu mô đun chương trình khác nhau yêu cầu?
- Xác định bao nhiêu vùng địa lý liên quan (chi nhánh văn phòng).
- Ước lượng số người dùng ứng dụng và thời gian ứng dụng được duy trì.
- Tính chính xác.
- Cuối cùng, hiểu khách hàng mong đợi dự án được triển khai.

Tại thời điểm này, chúng ta có ý tưởng phạm vi dự án. Cân nhắc độ lớn dự án đối với thời gian và ràng buộc ngân sách. Nếu dự án quá lớn về thời gian và tiền bạc cho chi trả thì bàn bạc vấn đề với khách hàng để đưa ra quyết định thương lượng cho thõa đáng. Chúng ta phải chọn lựa hoặc nhiều thời gian hơn, hoặc nhiều tiền hơn hoặc cả hai. Hoặc chúng ta phải giảm phạm vi dự án xuống. Phân tích tất cả những tình huống ở giai đoạn đầu của dự án sẽ làm cho dự án thành công nhiều hơn.

1.1.2 Phân tích mở rộng yêu cầu nghiệp vụ

a. Xác định yêu cầu nghiệp vụ

Mỗi dự án sẽ có một hay nhiều yêu cầu nghiệp vụ. Mỗi yêu cầu nghiệp vụ là một mô tả tác nhiệm cụ thể trong nghiệp vụ của khách hàng. Ví dụ. lưu vết quá trình đầu tư. Một tác vụ như kiểm soát đầu tư cần chia nhỏ thành những phần chắc chắn cho đến khi mỗi phần đủ để mô tả công việc chính xác

Khi mức độ của thành phần chia nhỏ dưới mức tối thiểu, xác định lại trình tự thành phần.

Mỗi tác vụ được gọi là yêu cầu nghiệp vụ hay quy tắc nghiệp vụ. Quy tắc doanh nghiệp được viết theo ngôn ngữ được hiểu bởi những người không chuyên máy tính sao cho người dùng có thể kiểm tra luật một cách chính xác

b. Xác định yêu cầu chất lượng khách hàng

Mỗi dự án phần mềm có thể yêu cầu nhanh, bảo mật, phụ thuộc, dễ dùng, hay bug-free. Trong thế giới thực, thời gian và ràng buộc tài chính làm cho không thể tạo ra những chương trình dự án hoàn chỉnh. Thay vào đó, điều quan trọng để quyết định dựa trên mức độ chấp nhận của chất lượng thõa mãn khách hàng.

Ví dụ: khi khách hàng quyết định ứng dụng phải sẵn sàng 23 giờ trong ngày, bỏ qua thời gian vận hành không giảm. Chất lượng khác bao gồm số người dùng truy cập hiện hành, thời gian tối đa phải chờ để hoàn thành công việc trong ứng dụng (sự phản hồi), độ bảo mật ứng dụng, hay hơn nữa.

c.Phân tích hạ tầng cơ sở hiện hành

Phần quan trọng trong thiết kế giải pháp là phân tích kỹ thuật thay thế. Điển hình, giải pháp phần mềm được đưa vào hơn là thay thế hệ thống hiện hành. Dự án cần làm việc trên phần cứng và phần mềm mà người dùng hiện có. Biết được hệ điều hành đang được cài trên máy của người dùng, loại mạng đang sử dụng, và nếu người dùng đang chạy phần mềm không tương thích với chương trình mới hơn. Nên bỏ thời gian tìm hiểu máy chủ hiện hành, hệ điều hành, phần mềm đang chạy.

Khi đưa giải pháp, nhớ rằng cơ sở hạ tầng hiện hành đảm bảo giải pháp của chúng ta có thể tương thích.

d. Phân tích ảnh hưởng kỹ thuật

Nếu cần mở rộng chức năng cho hệ thống hiện hành, chúng ta mong ước thay đổi hệ thống cũ cả việc cải thiện hệ thống cũ và tích hợp dễ dàng hơn hệ thống mới. Ví dụ, chức năng của chương trình kế toán lưu trữ dữ liệu nhỏ như CSDL hướng đến tập tin Access. Để tạo dữ liệu truy xuất hiệu quả hơn và thõa mãn yêu cầu của giải pháp mới, chúng ta mới chuyển toàn bộ dữ liệu sang hệ quản trị csdl SQL Server. Việc suy nghĩ trước sẽ tiết kiệm thời gian sau đó: trãi qua thời gian tìm hiểu sự khác biệt về giao tác, bảo mật, và những chức năng khác giữa kỹ thuật cũ và giải pháp mới.

Chúng ta nên tìm hiểu thủ tục chuyển đổi dữ liệu từ kỹ thuật cũ sang kỹ thuật mới. Đảm bảo được phép thực nghiệm những thủ tục này, và có kế hoạch bảo lưu trong trường hợp thực hiện vấn đề này bị lỗi. Đảm bảo chắc chắn những tác động chuyển đổi trên mọi thành phần của hệ thống, không chỉ phần tử gần nhất thay đổi.

1.1.3.Phân tích yêu cầu bảo mật

Khi hệ thống lưu trữ, truy xuất dữ liệu cá nhân như thông tin nhân sự, thẻ tín dụng, doanh số bán hay thông tin riêng tư, chúng ta cần có biện pháp đảm bảo an toàn những dữ liệu này.

a. Xác định vai trò

Toàn bộ ứng dụng không chỉ có 1 mức độ bảo mật. Người dùng cuối chỉ cần quyền truy xuất giới hạn vào hệ thống. Quản trị hệ thống, người thao tác viên cập nhật, và người dùng có quyền truy cập cao hơn ở mọi cấp độ. Bảo mật dựa trên vai trò là kỹ thuật dùng để cấp quyền mức độ bảo mật khác nhau tương ứng quyền hạn và độ chuyên nghiệp của mỗi người dùng trong hệ thống.

Lưu ý: Nhận biết những lớp chính của những người dùng cần truy cập đến ứng dụng của chúng ta. Gán tên vai trò cho mỗi lớp người dùng. Cuối cùng, gán mức độ tối thiểu có thể truy xuất đến mỗi vai trò. Mỗi lớp người dùng nên có đủ quyền truy xuất đến công việc của họ, và không nhiều hơn.

b. Xác định môi trường bảo mật ứng dụng

Độ bảo mật không bị giới hạn người dùng hệ thống. Chỉ người dùng đăng nhập vào ứng dụng, ứng dụnng phải "login" để kiểm soát tài nguyên chia sẻ như tập tin, dịch vụ hệ thống, cơ sở dữ liệu. Mức độ kiểm soát của ứng dụng được gọi là ngữ cảnh bảo mật. Chúng ta cần phải làm việc với nhiều người dùng khác như quản trị mạng, cấp quyền truy xuất phù hợp ứng dụng để chia sẻ tài nguyên.

c. Xác định ảnh hưởng bảo mật

Nếu công ty có sẵn cơ chế bảo mật thay vào đó hệ thống của chúng ta nên điều chỉnh cho phù hợp với cơ chế đã có. Nếu chúng ta đang thực thi hệ thống bảo mật mới hay một hệ thống khác, cần phải phân tích tác động của hệ thống trên hệ thống hiện tại:

- Hệ thống mới có làm hỏng chức năng của phần mềm hiện tại?
- Hệ thống đòi hỏi phải hỗ trợ thêm một phần người dùng – đăng nhập mở rộng ?
- Hệ thống sẽ khóa một vài người dùng trên những tập tin hay những tài nguyên mà họ được quyền truy cập trước đây

d. Kế hoạch vận hành

Khi tổ chức phát triển và thay đổi, người dùng mới được thêm vào, người cũ được cập nhật và bỏ đi. Những thao tác

này đòi hỏi thay đổi CSDL bảo mật, đó là nơi thông tin người dùng và quyền hạn truy cập của họ được lưu. Những thông tin này được lưu trữ hiện thời.

Nếu người dùng có vị trí địa lý khác nhau, ở văn phòng khác nhau, chúng ta cần lên kế hoạch tái tạo cơ sở dữ liệu bảo mật. Sự tái tạo là sự thay đổi hệ thống dữ liệu tại nơi này sao chép đến nơi khác sao cho tất cả thông tin bảo mật được lưu giữ mỗi nơi. Thuận lợi việc tạo bản sao là người dùng có thể đăng nhập dùng thông tin được lưu ở vị trí gần hơn so với vị trí địa lý. Nếu mạng WAN bị ngừng hoạt động, ví dụ người dùng vẫn có thể đăng nhập. Việc tạo bản sao cần được lên kế hoạch và vận hành.

Lưu ý: Chúng ta lên kế hoạch cho điều kiện khẩn cấp – phải làm gì nếu csdl bảo mật bị ngắt hay nếu việc tạo bản sao bị hỏng. Đối với hệ thống bảo mật bị hỏng, chúng ta cũng nên có cả hai kế hoạch khẩn cấp và thủ tục tự động chú ý đến những vấn đề chung như mạng bị hỏng.

d. Kế hoạch kiểm soát và đăng nhập

Một hệ thống bảo mật tốt không là cơ chế thụ động. Thay vào đó, chứa chức năng trợ giúp kiểm soát hoạt động của hệ thống cho vấn đề bảo mật. Vấn đề chung của chức năng này là nhật ký. Toàn bộ thao tác của hệ thống có thể được ghi nhận hầu như toàn bộ sự kiện liên quan đến bảo mật hệ thống. Có thể ghi nhận mỗi khi đăng nhập, truy xuất đến mọi tài nguyên nhưng điều này hiếm khi hiệu qủa; thường chúng ta sẽ ghi nhận một số tập thông tin này như việc cố gắng đăng nhập lỗi.

Lưu ý: Nhật ký hệ thống tự nó thì không có ý nghĩa; chúng ta phải kế hoạch kiểm soát thường xuyên bởi ta có thể phát hiện những nghi ngờ những mẫu nhật ký hoạt động. Người kiểm soát được huấn luyện nên phân tích nhật ký trên

cơ sở thường xuyên, đưa ra những đề nghị nếu có bất kỳ điều nghi ngờ.

e. Xác định mức độ yêu cầu bảo mật

Bảo mật cũng giống như những phần khác trong thiết kế ứng dụng, là sự cân nhắc giữa hiệu quả và chi phí. Nếu hệ thống không lưu những dữ liệu có tính nhạy cảm cao. Cách tốt nhất để triển khai hệ thống đó là "giữ sự xác thực của người dùng" đòi hỏi lưu trữ. Nếu chúng ta lưu trữ thông tin cần cho bảo mật, chi phí cho bảo mật thông tin đặc biệt phải được kiểm chứng.

Không có hệ thống nào bảo mật 100%. Chúng ta phải xác định mức độ rủi ro bảo mật có thể chấp nhận được. Độ rủi ro bảo mật diễn tả tỉ lệ phần trăm tương xứng khả năng mà bảo mật hệ thống không bao giờ đạt đến. Điều đó có thể nhưng phí tổn để xây dựng hệ thống bảo mật 99%. Chúng ta hay khách hàng phải xác định mức độ rủi ro có thể chấp nhận được dựa trên dữ liệu nhạy cảm của hệ thống.

f. Rà soát bảo mật hiện tại

Chúng ta nên trung thành ý tưởng của yêu cầu bảo mật của ứng dụng. Ở thời điểm phân tích chính sách bảo mật hiện tại của công ty để xác định bảo mật có đạt đến những nhu cầu của hệ thống hay không. Nếu không, thảo luận vấn đề với người gách vác hệ thống bảo mật ở công ty để tìm ra giải pháp mang lại lợi ích để triển khai mở rộng bảo mật.

1.1.4.Phân tích yêu cầu tốc độ

Tốc độ của ứng dụng có thể đòi hỏi khó. Đối với người dùng, ứng dụng sẽ hầu như chạy quá chậm nhưng chạy nhanh ứng dụng thiết kế tốt có thể mang lại giá trị

Lưu ý: việc chạy nhanh một ứng dụng thiết kế kém thì dễ, nhiều ứng dụng có thể chạy chậm bởi thiết kế thiếu sót,

những không bởi không tương thích giữa phần ứng và các yếu tố bên ngoài.

Chúng ta nên nhận thức yêu cầu tốc độ ứng dụng trước khi bắt đầu qui trình thiết kế. Yêu cầu tốc độ dựa theo các mục sau:

Mỗi phút giao dịch: cung cấp dịch vụ phụ thuộc vào số lượng lớn người dùng, ứng dụng phân tán dùng những giao tác. Số giao tác mỗi phút (TPM) là độ đo tốc độ hệ thống cơ sở dữ liệu.

Băng thông: Ứng dụng phân tán làm nghẽn việc sử dụng mạng. Sự phản hồi của ứng dụng xác định định băng thông mạng (độ rộng của đường truyền mạng). Băng thông thường được đo bằng megabit mỗi giây.

Khả năng chứa: Lượng lưu trữ- cả chính và phụ - sẵn sàng đối với ứng dụng là vấn đề lưu tâm quan trọng cho tốc độ chung của ứng dụng. RAM đòi hỏi của ứng dụng gây ra những khác biệt lớn cho tốc độ của ứng dụng.

Nút thắt: Trong mỗi hệ thống, có phần giới hạn tốc độ hệ thống nói chung. Ví dụ CPU tốc độ nhanh cũng không cải thiện gì mấy nếu phải chờ dữ liệu từ một ổ cứng quá chậm. Trong trường hợp này, ổ cứng sẽ là nút thắt của toàn bộ hệ thống. Không thể tăng tốc độ trừ khi nút thắc được nhận biết, bởi vì chỉ có cải thiện nút thắt làm nâng tốc độ phù hợp. Chúng ta có thể nhận biết nút thắt bằng cách sử dụng công cụ báo cáo hệ thống như Màn hình điều khiển tốc độ trên Window NT (Windows NT Performance Monitor).

Thuật ngữ tốc độ thường dùng đồng nghĩa với sự phản hồi - số lượng thời gian chiếm giữ để phản hồi lại hành động của người dùng. Có thể làm cho ứng dụng xuất hiện phản hồi mà không cần tăng tốc độ. Tuy nhiên, thời gian phản hồi trung bình của ứng dụng là đặc tính quan trọng, chúng ta phải kết

hợp chặt chẽ những mục tiêu thời gian phản hồi đối vời yêu cầu chung thiết kế.

Không thể nói về tốc độ trong những ứng dụng phân tán mà không phân biệt quan trọng: giữa nhu cầu cao và trung bình. Tại một số thời điểm - tối hay cuối tuần – có lẽ ứng dụng sẽ phục vụ với số lượng nhỏ người dùng, thì tốc độ nó sẽ trên trung bình. Ở thời điểm khác, số lượng người dùng sẽ cao hơn và tốc độ ứng dụng đủ cho phép. Mục tiêu tốc độ bao gồm cả mục tiêu tốc độ trung bình và cao.

1.1.5 Phân tích yêu cầu vận hành

Chúng ta có thể giảm bớt chi phí vận hành theo nhiều cách.Cách tốt nhất để giảm chi phí vận hành là đảm bảo chương trình được kiểm thử và chạy debug trước khi đưa vào triển khai. Chi phí triển khai có thể được giảm bớt bởi phân phối trực tuyến hay những thủ tục tự động cài đặt, và qui trình vận hành có thể tự động bằng các qui trình tin học. Mức vị trí và huấn luyện đội ngũ là vấn đề xem xét quan trọng: đội ngũ nhân viện càng được huấn luyện kỹ và sâu thì vấn đề càng nhanh chóng được sửa đổi.

Trong trường hợp phần cứng, phần mềm là thành phần được mua chứ không được phát triển, chúng ta có thể nhận sự chấp thuận vận hành từ nhà xưởng hay người ủy thác của sản phẩm. Vận hành sản phẩm trung gian tiết kiệm cho chúng ta chi phí thuê mướn nhân viên mới hay huấn luyện lại những nhân viên cũ để duy trì một hay nhiều thành phần của hệ thống.

Giảm chi phí vận hành đòi hỏi sự tự thõa mãn lợi nhuận trong thời ngắn đối với những lợi ích trong tương lai. Giảm chi phí vận hành lâu dài thường đòi hỏi đầu tư đón đầu trong tự động hóa phần cứng và phần mềm.

1.1.6 Phân tích khả năng mở rộng yêu cầu

Qua thời gian, những yêu cầu giải pháp sẽ thay đổi. Người dùng cần những chức năng mới, các quy luật đặt ra sẽ bị sửa đổi, và phần cứng phần mềm nền mới thay đổi theo. Ứng dụng thiết kế tốt là có khả năng mở rộng được – nó có thể uyển chuyển cải thiện mà không phải viết lại hoàn toàn. Khả năng mở rộng của ứng dụng bị đảo ngược so với lượng công việc cần hoàn thành để thêm những đặc trưng mới.

Khả năng mở rộng có thể đạt được thông qua những ý nghĩa khác nhau. Một cách đạt những khả năng hạn định là lưu trữ thông tin quy luật đặt ra trong cơ sở dữ liệu hơn là lập trình chúng trong đối tượng nghiệp vụ. Theo cách đó, nếu số quan trong hay thủ tục thay đổi, nó có thể thay đổi trong CSDL mà không thay đổi mã nguồn. Cách khác là đặt mã nguồn vào trong đoạn script được làm rõ hơn biên dịch chương trình; đoạn script có thể bị thay đổi một cách dễ dàng không đòi hỏi bất kỳ biên dịch hay cài đặt lại tập tin nhị phân

Lưu ý: cách tốt nhất để đạt được khả năng mở rộng là ngắt ứng dụng thành những đối tượng thành phần, mỗi thành phần hoàn thành một nhiệm vụ riêng lẻ. Nếu những yêu cầu của những nhiệm vụ đặc biệt thay đổi, đối tượng tương ứng có thể bị thay đổi và biên dịch lại mà không gây ảnh hưởng bất kỳ đối tượng khác. Những đối tượng được thêm vào dễ dàng. Đối tượng nghiệp vụ có những thuận lợi được làm hiệu quả hơn những phương pháp khác trong khi vẫn đảm bảo tốt khả năng mở rộng.

1.1.7. Phân tích những yêu cầu sẵn có

Những ứng dụng phân tán được thiết kế để chạy mỗi ngày. Nó cần thiết cho sự thành công của doanh nghiệp. Như vậy, chúng có mức độ sẵn sàng cao nên tránh thường bảo trì, sửa chữa, phát sinh không theo kế hoạch.

Rõ ràng, đối với những ứng dụng mang tính sẵn sàng, nó không được gây ra lỗi. Không có ứng dụng nào là không có lỗi, ứng dụng phải được bảo lưu để chúng có thể hoạt động thậm chí khi bug xảy ra trong một phần của chương trình. Thí dụ, nếu người dùng gây ra lỗi cho chương trình thì chỉ một phần chương trình phục vụ cho người dùng đó bị hỏng, không ảnh hưởng người dùng còn lại đang nối kết. Bất kỳ thành phần ứng dụng nào hỏng hay không sẵn sàng thì nên khởi động lại ngay khi có thể.

Việc bảo trì có kế hoạch cũng tác động đến tính sẵn sàng. Một máy chủ chứa ứng dụng lý tưởng luôn có bản sao lưu có thể khởi động khi máy chủ bảo trì. ứng dụng có mức độ sẵn sàng cao có cách luân phiên để kết nối mạng trong trường hợp mạng WAN, LAN ngưng hoạt động

Lưu ý: Tính sẵn sàng liên quan đến nghiệp vụ. Tính sẵn sàng của ứng dụng càng cao, giá trị của ứng dụng càng cao. Chúng ta phải xác định bao nhiêu giờ trong ngày ứng dụng cần được thao tác; giờ nào là quan trọng so với các giờ trong ngày. Cân nhắc giá trị của việc tăng tính sẵn sàng đối với giá trị dự án của thời gian down ứng dụng. Những hệ thống trọng yếu, giá trị đối với công ty ở bất kỳ thời điểm down nào hoàn toàn điều chỉnh chi phí thiết kế 100 % ứng dụng sẵn sàng. Ứng dụng khác đơn giản cần trở nên sẵn sàng hầu hết mọi lúc.

1.1.8. Phân tích yêu tố con người

Thiết kế ứng dụng được giám sát bởi nhiều người lập trình là phần quan trọng của yếu tố con người. Chúng ta nên xác định kinh nghiệm gì mà chúng ta muốn người dùng có. Với bất cứ ứng dụng nào khác, kinh nghiệm người dùng càng tốt thì chi phí càng cao.

Bắt đầu định nghĩa mục tiêu của người dùng. Xác định người dùng với những nhu cầu đặc biệt như thế nào. Chúng ta

cần điều tiết người dùng qua việc điều tiết nghe và nhìn, hay người dùng nói tiếng nước ngoài. Phụ thuộc vào vị trí địa lý của người sử dụng. Chúng ta cần sửa đổi ứng dụng thích ứng theo vị trí địa lý. Cần điều chỉnh nhu cầu lướt qua của người dùng, người không cần sự nối kết chắc chắn hay khả năng trả lời lại.

Xem xét mức độ chuyên nghiệp giữa người dùng. Với chuyên viên học nhanh hơn với giao diện thiết kế tốt và trợ giúp trực tuyến Help online. Người dùng với kỹ năng kém hơn để tăng tốc qua sử dụng wizard, trợ giúp online, hay chỉ dẫn. Huấn luyện khách hàng trong ứng dụng cũng nên cân nhắc chọn lựa.

1.1.9. Phân tích yêu cầu tích hợp

Nếu giải pháp giao tiếp với ứng dụng kế thừa, việc truy xuất CSDL tồn tại, hay việc chuyển đổi dữ liệu cũ sang khuôn dạng mới, bạn cần phải đưa kế hoạch tích hợp ứng dụng với phần mềm cũ. Điều này được làm thông qua kết nối của hãng trung gian như trình điều khiển thiết bị kết nối csdl (ODBC), nhưng chúng ta cũng cần viết kết nối và những tiện ích chuyển đổi

Khi phát sinh nhu cầu lớn hơn, cơ sở dữ liệu phải thiết kế lại. Kỹ thuật dữ liệu mới hay vậ lý đưa nhu cầu cải thiện CSDL bên dưới ứng dụng. Những cải tiến phải được cẩn thận bởi chúng phá vở tất cả mã nguồn CSDL hiện tại. Trước khi cải tiến khung dữ liệu, đảm bảo những phần mã nguồn hiện tại có thể truy xuất đến CSDL. Tất cả mã nguồn hiện tại phải được soát lại, có thể viết lại.

1.1.10. Phân tích thực tiễn nghiệp vụ tồn tại

Phần định nghĩa trong qui tắc nghiệp vụ liên quan đến sự hiểu biết ngữ cảnh trong những qui tắc thao tác. Hiểu được

những thực tế nghiệp vụ của doanh nghiệp có thể giúp chúng ta tránh được sai sót thậm chí giúp tìm cách tốt hơn, hiệu quả hơn của tự động hóa tiến trình nghiệp vụ. Hiểu được vấn đề hợp lệ dưới mỗi tiến trình có thể ngăn bạn gây ra lỗi một cách ngây ngô dẫn đến tranh chấp.

Hiểu được cấu trúc tổ chức và sơ đồ làm việc nghiệp vụ là quyết định. Không hiểu rõ ràng sơ đồ tổ chức, không thể đem lại sự chấp thuận phù hợp cho thiết kế ứng dụng của chúng ta hay thông tin theo kíp trên thiết kế hay những vấn đề triển khai. Đồ hình tổ chức cũng giúp cho tìm kiếm thông tin người ẩn danh phản hồi lại chức năng của ứng dụng mà không dùng bất của chính họ.

Có được ứng dụng từ giai đoạn phát triển đến sản phẩm đòi hỏi sự hiểu biết mạng và chính sách hạ tầng của công ty. Biết được ai là người chịu trách nhiệm bảo trì, bảo mật, tính toàn vẹn, khả năng phản hồi tương tác trên mạng. Học những tiến trình và chính sách liên quan chạy trên ứng dụng mới. Tìm ra loại kiểm soát chất lượng và dịch vụ kiểm thử sẵn sàng trong khi chúng ta kiểm thử trên chính phần mềm, ta có thể tự động tài nguyên hay dành cho bộ phận kiểm tra chất lượng tùy ý sử dụng. Chúng ta có thể yêu cầu phương pháp thiết kế đặc biệt hay triển khai thực tế. Chúng at cũng đòi hỏi chắc chắn kế hoạch được kết chặt với ngân sách

Cuối cùng, giữ những nguyên tắc cốt lỏi: Học nhu cầu khách hàng, cố gắng thực hiện chúng. Điều này có thể trở nên khó khi khách hàng không biết nhu cầu của họ là gì, nhưng đó là cách dẫn đến ứng dụng thành công.

1.1.11.Phân tích yêu cầu khả năng quy mô

Nếu ứng dụng thành công sẽ hấp dẫn người dùng hơn. Đặc biệt, nếu ứng dụng chạy trên môi trường web như Internet thì sự thành công đồng nghĩa với tăng nhu cầu. Ứng dụng phải

được thiết kế có quy mô- nó phải hỗ trợ nâng cấp cho phép phục vụ nhiều người hơn.

Một cách đơn giản để nâng cao ứng dụng là mua CPU nhanh hơn, nhiều RAM, kết nối mạng tốt hơn. Tuy nhiên việc tăng cường máy đơn chạy nhanh hơn. Thực sự những ứng dụng có thể nâng cấp phải thêm vào nhiều dịch vụ phía máy chủ. Điều này có nghĩa ứng dụng có thể chạy trên nhiều máy tính cùng một lúc, sự phân phối việc tải xuống của người dùng và xử lý thời gian qua nhiều máy chủ. Điều này sẽ gia tăng đáng kể tính phức tạp, vì vậy một lần nữa tính thuận tiện khả năng quy mô phải được cân nhắc đối với giá trị cung cấp. Tuy nhiên, ứng dụng như Miscrosoft Transaction Server giảm đáng kể chi phí phát triển ứng dụng phân tán bởi quản lý về mặt logic của phân tán tự động.

1.2 Xác định yêu cầu

Mục tiêu của việc xác định yêu cầu*:*

Xác định thật chính xác và đầy đủ các yêu cầu đặt ra cho phần mềm sẽ được xây dựng.

Kết quả nhận được sau giai đoạn xác định yêu cầu:

1. *Danh sách các công việc sẽ được thực hiện trên máy tính*
2. *Những mô tả chi tiết về các công việc này khi được thực hiện trong thế giới thực***.**

Qua đó bước đầu hình thành thông tin khái quát về các hoạt động trong thế giới thực.

1.2.1 Yêu cầu và mô tả yêu cầu

- Yêu cầu (hay yêu cầu phần mềm) là công việc muốn thực hiện *trên máy tính*. Những công việc này phải xuất phát từ thực tế chứ không thuần túy tin học
- Mô tả yêu cầu là mô tả đầy đủ các thông tin liên quan đến công việc tương ứng. Các mô tả này dùng làm cơ sở để nghiệm thu và đánh giá phần mềm khi được chuyển giao.

Các yêu cầu của phần mềm cần được mô tả thật rõ ràng, cụ thể, đầy đủ và chính xác các thông tin liên quan đến công việc tương ứng. Việc mô tả sơ sài, mơ hồ yêu cầu phần mềm sẽ dẫn đến việc hiểu nhầm giữa chuyên viên tin học (người thực hiện phần mềm) và khách hàng (người đặt hàng thực hiện phần mềm). Nhiều công sức và chi phí phải hao tốn do các hiểu nhầm như thế.

Các loại thông tin chính cần được quan tâm khi xác định yêu cầu phần mềm:

- Tên công việc ứng với từng yêu cầu
- Người hoặc bộ phận sẽ thực hiện công việc
- Địa điểm thực hiện công việc
- Thời gian thực hiện công việc
- Cách thức tiến hành công việc cùng với các quy định liên quan

Sau đây, từng loại thông tin sẽ lần lượt được xem xét chi tiết:

a. ***Tên công việc.***

Cần xác định cụ thể, tránh dùng các tên chung chung, mơ hồ

Ví dụ: xét một số tên công việc sau:

Quản lý độc giả: chung chung, mơ hồ; cụ thể như việc đăng ký mượn sách, gia hạn thẻ độc giả, trả sách

Quản lý sách: chung chung, mơ hồ; cụ thể như nhập sách vào kho, tra cứu sách, cho mượn sách, nhận trả sách, thanh lý sách.

b. ***Người thực hiện.***

Cần xác định chính xác người hoặc bộ phận sẽ thực hiện công việc trên máy tính (còn gọi là người dùng phần mềm hay người dùng).

Những người dùng có vai trò và công việc thực hiện tương tự như nhau sẽ được xếp vào cùng một loại người dùng (thông thường một loại người dùng sẽ tương ứng với một bộ phận trong thế giới thực).

Cùng một công việc có thể có nhiều loại người dùng khác nhau thực hiện và ngược lại, một loại người dùng có thể thực hiện nhiều công việc khác nhau.

c. ***Thời gian, địa điểm.***

Cần xác định chính xác địa điểm, thời điểm tiến hành công việc. Các thông tin này sẽ có ý nghĩa nhất định trong một số trường hợp đặc thù.

d. ***Cách thức tiến hành và các quy định liên quan.***

Đây là phần chính yếu khi tiến hành mô tả yêu cầu. Đối với loại thông tin này cần đặc biệt quan tâm đến một số yếu tố sau:

i. Các quy định cần kiểm tra khi thực hiện công việc ghi nhận thông tin

Ví dụ: Quy định về việc mượn sách khi cho độc giả mượn sách: chỉ cho mượn sách đối với những độc giả có thẻ

độc giả còn hạn, số sách đang mượn chưa đến 2 và không có sách mượn quá hạn.

Ví dụ: Quy định tính hợp lệ của phân số trong việc ghi nhận đề bài của giáo viên và bài giải của học sinh: phân số phải có mẫu số khác 0

ii. Các quy định, công thức tính toán khi thực hiện công việc tính toán

Ví dụ: Quy định tính tiền phạt trả sách trễ khi thực hiện việc trả sách: mỗi ngày trả trễ phạt 1500 đồng/ngày. Từ ngày trả trễ thứ 10 trở đi sẽ phạt 5000 đồng/ngày và thu hồi thẻ độc giả 2 tuần.

Ví dụ: Quy định tiền lương khi thực hiện công việc tính lương nhân viên cho 1 công ty

* Lương của nhân viên thuộc bộ phận văn phòng được tính theo công thức:
Tiền_Lương = (Số_Ngày * Mức_Lương)/22 + Tiền_Thưởng

+ Tiền_Phạt
mỗi ngày làm thêm thưởng 30.000
mỗi ngày nghỉ việc phạt 50.000

* Lương của nhân viên thuộc bộ phận sản xuất được tính theo công thức:

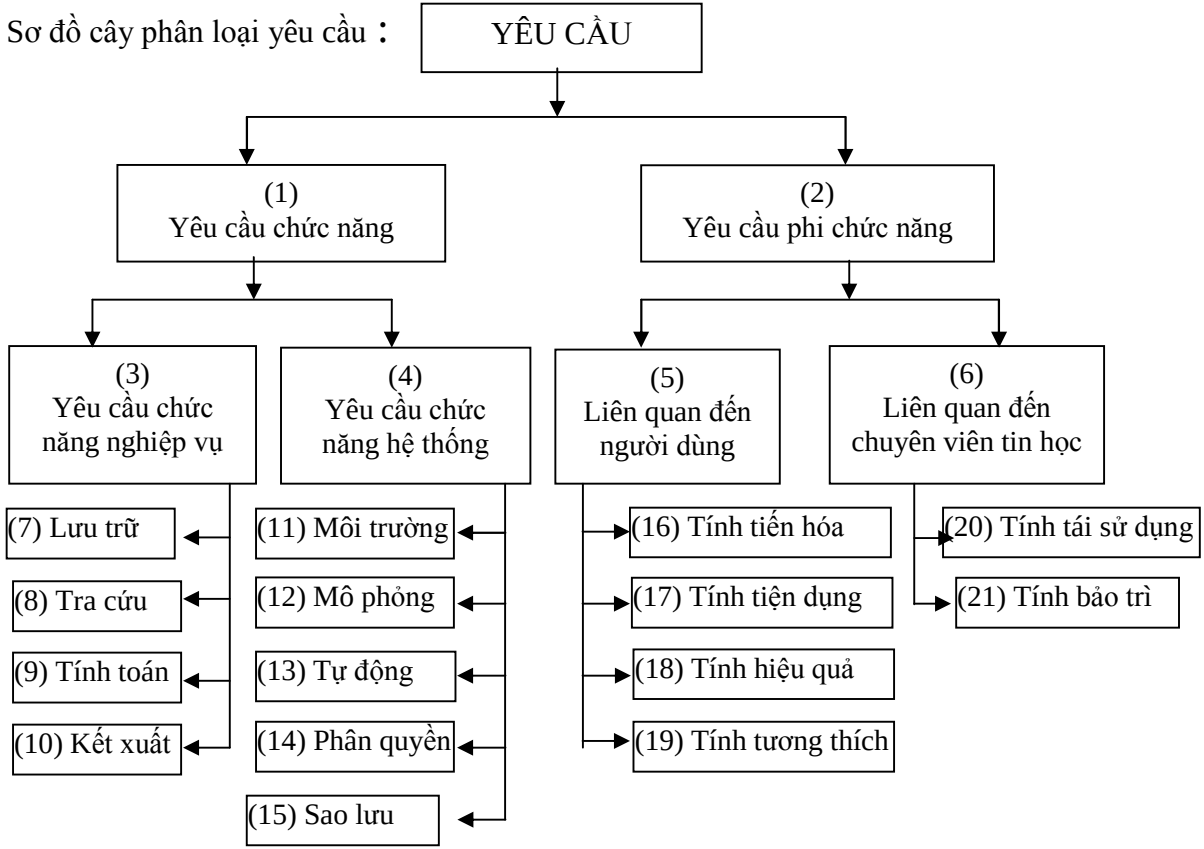
Tiền_Lương = Số_Sản_Phẩm * Đơn_Giá

Biết rằng một sản phẩm phải trải qua 3 công đoạn sản xuất:

công đoạn 1: 200 đồng/sản phẩm
công đoạn 2: 400 đồng/sản phẩm
công đoạn 3: 300 đồng/sản phẩm

1.2.2 Phân loại yêu cầu

Sơ đồ cây phân loại yêu cầu：



Đặc tả chi tiết từng loại yêu cầu:

(1) Yêu cầu chức năng là danh sách các công việc sẽ được *thực hiện trên máy tính* cùng với các thông tin mô tả tương ứng.

(2) Yêu cầu phi chức năng là các yêu cầu liên quan đến chất lượng phần mềm, là sự ràng buộc cách thức thực hiện các yêu cầu chức năng.

(3) Yêu cầu chức năng nghiệp vụ là các chức năng của phần mềm tương ứng với công việc có thật trong thế giới thực.

(4) Yêu cầu chức năng hệ thống là các chức năng phần mềm *được phát sinh thêm* khi thực hiện công việc trên máy tính thay vì trong thế giới thực hoặc các chức năng không tương ứng với bất kỳ công việc nào trong thế giới thực.

(5) Chức năng lưu trữ: Tương ứng với công việc ghi chép thông tin trên sổ sách (kèm theo các quy định khi ghi chép).

Ví dụ:

- Ghi nhận việc cho mượn sách của một thư viện theo quy định mượn.
- Ghi nhận bài giải bài tập về phân số theo quy định về phân số,cách biến đổi phân số tương đương, các phép tính trên phân số,…

(6) Chức năng tra cứu: Tương ứng với công việc tìm kiếm, theo dõi hoạt động và xem thông tin về một đối tượng.

Ví dụ:

- Tìm tài khoản và xem tình hình gửi rút.
- Tìm sách và xem tình trạng sách
- Tìm hàng hóa và xem tình trạng của hàng hóa (số lượng tồn kho, lượng nhập, thời gian nhập,).
- Tìm bài giảng lý thuyết về phương trình, bất phương trình và xem nội dung tương ứng.

(7) Chức năng tính toán: Tương ứng với công việc tính toán (theo quy định và công thức cho trước).

Ví dụ:

- Tính điểm trung bình môn học của học sinh theo quy định hệ số cho các đợt kiểm tra.
- Xếp thứ hạng cho các đội bóng sau một lượt thi đấu theo quy định của ban tổ chức giải.
- Tính tiền phạt trả sách trễ theo quy định phạt của thư viện.
- Tìm nghiệm của phương trình bậc hai theo phương pháp giải phương trình bậc hai.

(8) Chức năng kết xuất : Tương ứng với công việc lập báo cáo (theo biểu mẫu cho trước)

Ví dụ:

- Lập bảng xếp hạng các đội bóng sau một lượt đấu.
- Lập báo cáo thống kê về số lượt mượn sách theo từng thể loại trong năm.
- Lập báo cáo thống kê về tỷ lệ xếp loại học sinh theo từng lớp, từng khối.

(9) Chức năng môi trường : Định cấu hình thiết bị, ngày giờ, số người làm việc, …

Ví dụ: Số lượng người làm việc, chọn loại máy in, khổ giấy, niên khóa hiện hành, …

(10) Chức năng mô phỏng: Mô phỏng hoạt động của thế giới thực

Ví dụ: - Mô phỏng một tai nạn máy bay, xe ô tô, trận động đất

(11) Chức năng tự động: Tự động thông báo, nhắc nhở người dùng.

Ví dụ:

- Nhắc nhở thủ thư gửi giấy báo đòi sách khi có độc giả mượn quá hạn.
- Báo động khi khách hàng thiếu nợ quá lâu hay số tiền nợ quá lớn.

(12) Chức năng phân quyền : Phân quyền sử dụng giữa các loại người dùng.

Ví dụ: Phân quyền cho 3 loại người sử dụng trong phần mềm quản lý thư viện:

+ Quản trị hệ thống: có quyền sử dụng tất cả các chức năng.
+ Thủ thư: chỉ sử dụng các chức năng liên quan đến việc cho mượn và trả sách.
+ Độc giả: chỉ sử dụng chức năng tra cứu.

Trong phần mềm quản lý bán hàng, việc phân chia khả năng truy cập dữ liệu nhập xuất cho từng nhóm người sử dụng sẽ tránh việc điều chỉnh số liệu không thuộc phạm vi quản lý của người sử dụng như nhân viên thu ngân chỉ được phép lập và điều chỉnh các hóa đơn bán hàng trong ca làm việc của mình. Ca trưởng và bộ phận quản lý quầy có thể tham khảo lượng hàng tồn kho nhưng không được phép điều chỉnh lượng hàng nhập, không được tham khảo vốn hàng xuất, kết quả kinh doanh, …

(13) Chức năng sao lưu : Sao lưu, phục hồi dữ liệu.

Ví dụ: Sao lưu thông tin về các học sinh đã ra trường và chỉ phục hồi lại khi cần thiết

(14) Tính tiến hóa: đây là các yêu cầu liên quan đến việc cho phép người dùng thay đổi lại cách mô tả của một yêu cầu chức năng (các quy định, quy tắc tính toán), một biểu mẫu nào đó khi đang dùng phần mềm đã được chuyển giao. Điều này đòi hỏi phải có dự kiến về các thay đổi trên thành phần dữ liệu và xử lý.

Ví dụ:

- Cho phép thay đổi quy định về số sách cho mượn tối đa, hay mức phạt khi trả trễ.

- Cho phép thay đổi các biên trong quy định về xếp loại học sinh.

(15) Tính tiện dụng: là các yêu cầu liên quan đến hình thức giao diện của phần mềm, thể hiện ở sự tự nhiên, dễ sử dụng, dễ học, đầy đủ thông tin,...

Ví dụ:

- Giao diện nhập hóa đơn bán hàng dạng form, dòng nhập thể hiện bằng ô sáng và báo lỗi khi số liệu nhập làm số lượng tồn kho âm (phần mềm quản lý hàng hóa).

(16) Tính hiệu quả : đây là yêu cầu liên quan đến thời gian thực hiện các chức năng phần mềm, dung lượng lưu trữ, chi phí sử dụng tài nguyên hệ thống như sử dụng tối ưu các không gian, thao tác thực hiện nhanh ...

Ví dụ: Thời gian tra cứu sách, tra cứu độc giả không quá 10 giây.

(17) Tính tương thích: là các yêu cầu liên quan đến việc chuyển đổi dữ liệu giữa phần mềm đang xét và các phần mềm khác, sự nhất quán giữa các màn hình trong hệ thống.

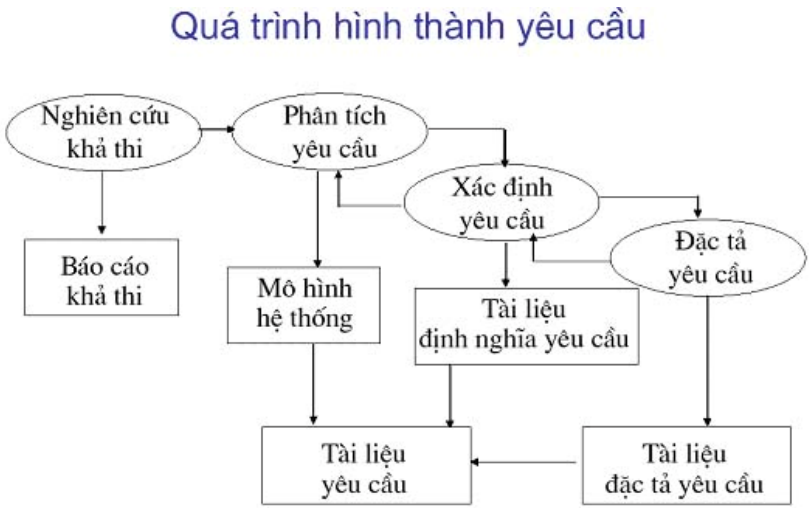
Ví dụ: - Cho phép chuyển tất cả các báo cáo sang định dạng file Excel

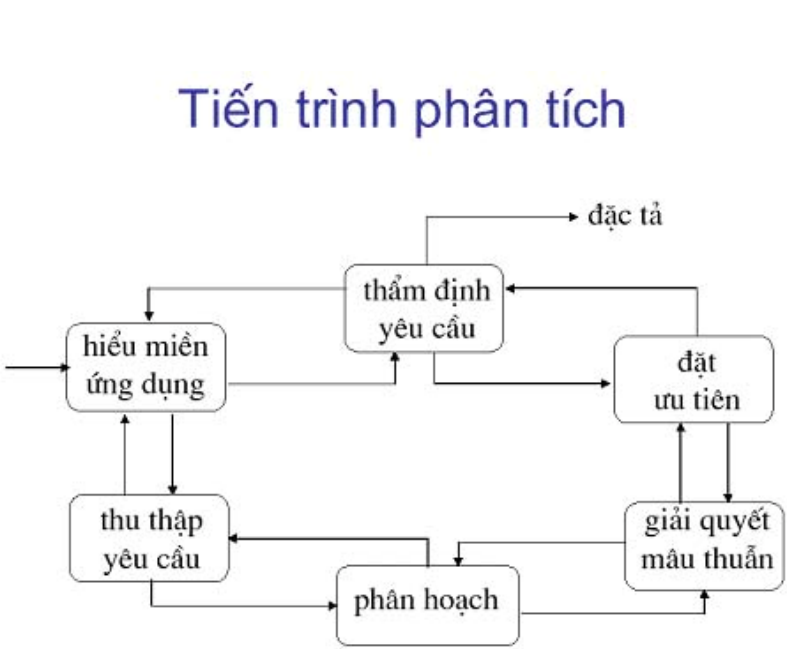
- Cho phép nhập thông tin sách mới từ tập tin Excel hay từ thiết bị đọc mã vạch.

- Cho phép thực hiện chức năng bằng giọng nói.

(18) Tính tái sử dụng: (do chuyên viên tin học đảm trách)

(19) Tính bảo trì: (do chuyên viên tin học đảm trách) là các yêu cầu cho phép thay đổi mà không làm ảnh hưởng đến phần mềm





1.2.3 Các bước xác định yêu cầu

Quá trình thực hiện xác định yêu cầu: gồm hai bước chính như sau

Bước 1: Khảo sát hiện trạng, kết quả nhận được là các báo cáo về hiện trạng.

Bước 2: Lập danh sách các yêu cầu, kết quả nhận được là danh sách các yêu cầu sẽ được thực hiện trên máy tính.

Đối tượng tham gia xác định yêu cầu: gồm hai nhóm người:

- *Chuyên viên tin học*: những người hiểu rõ về khả năng của máy tính. Họ phải tìm hiểu thật chi tiết về công việc của nhà chuyên môn nhằm tránh sự hiểu nhầm cho những bước phân tích sau này.
- *Nhà chuyên môn*: những người hiểu rõ về công việc của mình. Họ cần lắng nghe ý kiến của các chuyên viên tin học để đảm bảo các yêu cầu của họ là có thể thực hiện được với chi phí và thời gian hợp lý.

Hai nhóm người này cần phải phối hợp thật chặt chẽ để có thể xác định đầy đủ và chính xác các yêu cầu.

Sau đây, chúng ta sẽ phân tích chi tiết từng bước quy trình thực hiện.

1.2.3.1 Khảo sát hiện trạng

Các chuyên viên tin học sẽ tìm hiểu hiện trạng về các công việc của các nhà chuyên môn.

a. Các hình thức thực hiện phổ biến:

- Quan sát: theo dõi các hoạt động đang diễn ra ở thế giới thực có liên quan, có thể tiến hành ghi âm, ghi hình đối với những tình huống mang tính phức tạp, quan trọng, cần sự chính xác cao.

  Ví dụ:
  - Ghi hình quá trình giao dịch của một nhân viên ngân hàng với khách hàng tại một ngân hàng X.
  - Quan sát thao tác cho mượn sách của một thủ thư tại một thư viện Y

- Phỏng vấn trực tiếp: tổ chức phỏng vấn bắt đầu từ cấp lãnh đạo dần xuống các vị trí công việc. Có thể sử dụng các bảng câu hỏi có sẵn các câu trả lời cho đối tượng được phỏng vấn lựa chọn, …

- Thu thập thông tin, tài liệu: các công thức tính toán, quy định; các bảng biểu, mẫu giấy tờ có ít nhiều liên quan.

  Ví dụ:

  - Mẫu hóa đơn và các quy định lập hóa đơn bán hàng tại một cửa hàng Y.
  - Phiếu mượn sách tại thư viện của trường đại học Z.

b.Quy trình thực hiện:

- Tìm hiểu tổng quan về thế giới thực: bao gồm

- Quy mô hoạt động.
- Các hoạt động mà đơn vị có tham gia.

- Tìm hiểu hiện trạng tổ chức (cơ cấu tổ chức)

  Người tiến hành khảo sát hiện trạng cần hiểu rõ cơ cấu tổ chức các bộ phận của thế giới thực, đặc biệt là 2 yếu tố: trách nhiệm và quyền hạn. Sự hiểu rõ cơ cấu tổ chức giúp xác định bộ phận nào sẽ sử dụng phần mềm để có thể lên kế hoạch tiếp tục khảo sát chi tiết hơn bộ phận đó.

  Cơ cấu tổ chức bao gồm:

  - Đối nội.
  - Đối ngoại.
  - Các chức danh (Ví dụ: nhân viên nhập liệu, thủ thư, nhân viên bán hàng, …).

  Sử dụng các đồ hình để vẽ lại cơ cấu tổ chức.

- Tìm hiểu hiện trạng nghiệp vụ

  Thường diễn ra tại các vị trí công việc. Với bộ phận được chọn khảo sát chi tiết, người thực hiện khảo sát cần lập danh sách các công việc mà bộ phận này phụ trách, sau đó tìm hiểu các thông tin chi tiết cho từng công việc (thông tin mô tả yêu cầu phần mềm).

  Việc tìm hiểu dựa trên các ý sau:

  - Thông tin đầu vào.
  - Quá trình xử lý.
  - Thông tin kết xuất.

  Sau đó tiến hành xếp loại các nghiệp vụ vào 4 loại sau nhằm tránh thiếu xót khi tìm hiểu các thông tin:

  - Nghiệp vụ lưu trữ.
  - Nghiệp vụ tra cứu.
  - Nghiệp vụ tính toán.

- Nghiệp vụ tổng hợp, thống kê

1.2.3.2 Lập danh sách các yêu cầu

Để có được danh sách đầy đủ và chính xác các, quá trình lập danh sách các yêu cầu cầu theo các bước sau:

- Xác định yêu cầu chức năng nghiệp vụ
- Xác định yêu cầu chức năng hệ thống
- Xác định yêu cầu phi chức năng

a. Xác định yêu cầu chức năng nghiệp vụ.

Cách tiến hành: Nhà chuyên môn đề xuất và chuyên viên tin học sẽ xem xét lại

Bước tiến hành :

1. Xác định bộ phận (người dùng) sẽ sử dụng phần mềm
2. Xác định các công việc mà người dùng sẽ thực hiện trên phần mềm theo từng loại công việc sau:
   - Lưu trữ
   - Tra cứu
   - Tính toán
   - Kết xuất

Lần lượt lập bảng yêu cầu chức năng nghiệp vụ, bảng quy định/Công thức và các biểu mẫu – được mô tả chi tiết – như sau:

*Mẫu 1: Bảng yêu cầu chức năng nghiệp vụ

Bộ phận (người thực hiện): …
Mã số: ...

STT	**Công việc**	**Loại công việc**	**Quy định/ Công thức liên quan**	**Biểu mẫu liên quan**	**Ghi chú**
1					
2					

* Mẫu 2: Bảng Quy định/ Công thức liên quan

stt	Mã số	Tên Quy định/ Công thức	Mô tả chi tiết	Ghi chú
1	QĐ 1			
2	QĐ 2			

* Các biểu mẫu được mô tả chi tiết ngay sau bảng quy định/Công thức

Ví dụ: Xét phần mềm quản lý thư viện

Bộ phận: Thủ thư. **Mã số: TT**

stt	Công việc	Loại công việc	Quy định/Công thức liên quan	Biểu mẫu liên quan	Ghi chú
1	Cho mượn sách	Lưu trữ	TT_QĐ 1	TT_BM 1	
2	Nhận trả sách	Lưu trữ	Chỉ nhận lại những sách đã cho mượn	TT_BM 1	
3	Tính tiền phạt	Tính toán	Mỗi ngày trả trễ phạt : - 1000 đồng/ngày : từ ngày thứ nhất đến ngày thứ 5 - 3000 đồng/ngày : từ ngày thứ 6 trở đi.		
4	Tính tiền đền	Tính toán	Tiền đền cho sách bị mất dựa trên giá thị trường tại thời điểm hiện hành.		

5.	Tra cứu sách	Tra cứu	Việc tìm sách dựa trên các thông tin : tên sách, tên tác giả, nhà xuất bản, năm xuất bản		
6.	Gửi giấy báo đòi sách	Kết xuất	Sách mượn quá hạn 3 ngày sẽ tự động gửi giấy báo cho đến khi sách được trả hoặc đã tính xong tiền đền sách	TT_BM 2	

Bảng yêu cầu chức năng nghiệp vụ

stt	Mã số	Tên Quy định/ Công thức	Mô tả chi tiết	Ghi chú
1	QĐ 1	Quy định cho mượn sách	Chỉ cho mượn sách khi : - Thẻ độc giả còn hạn - Độc giả chưa mượn hết số sách quy định - Độc giả không có sách mượn quá hạn - Sách hiện không có người mượn	Độc giả mượn sách sẽ phải gửi lại thẻ độc giả tại bộ phận bạn đọc, nhận phiếu mượn sách (TT_BM 1, tìm kiếm mã số sách mượn và điền các sách cần mượn vào phiếu, xong gửi cho thủ thư.

Bảng Quy định/ Công thức liên quan

TT_BM 1:

PHIẾU MƯỢN SÁCH

Số thẻ: Số phiếu mượn:

Họ và tên: Ngày mượn:

[ ] Mượn về nhà [ ] Đọc tại chỗ

STT	Mã sách	Tên sách	Tác giả	Mã loại
1				
2				

Ngày ... tháng ... năm ...

TT_BM 2:

GIẤY BÁO MƯỢN SÁCH QUÁ HẠN

Thân gửi: _____

Địa chỉ: _____

Chúng tôi xin thông báo rằng, anh (chị) đã mượn của thư viện chúng tôi những quyển sách sau:

STT	Mã sách	Tên sách	Ngày mượn	Đến hôm nay đã quá hạn
1				
2				

Vậy thông báo anh(chị) vui lòng đem sách đến trả. Và mang theo số tiền _____ đồng để trả phí sách trễ.

Bộ phận: Độc giả. **Mã số: ĐG**

stt	Công việc	Loại công việc	Quy định/ Công thức liên quan	Biểu mẫu liên quan	Ghi chú
1	Tìm sách	Tra cứu	Việc tìm sách dựa trên các thông tin: tên sách, tên tác giả, nhà xuất bản, năm xuất bản		
2	Đăng ký mượn sách	Lưu trữ	Độc giả phải có thẻ độc giả.	TT_BM 1	Mọi độc giả có thẻ mượn sách đều có thể đăng ký mượn sách. Tuy nhiên, hệ thống sẽ thông báo khi thẻ mượn sách của độc giả đã hết hạn sử dụng.

Bộ phận: Quản lý độc giả. **Mã số : QLĐG**

stt	Công việc	Loại việc	Quy định/ Công thức liên quan	Biểu mẫu liên quan	Ghi chú
1	Làm thẻ độc giả mới	Lưu trữ	Chỉ cấp thẻ độc giả có độ tuổi từ 18 trở lên và có chứng minh thư. Lệ phí làm thẻ độc giả là 5000 đồng/thẻ. Một số chứng minh thư chỉ có thể có duy nhất một thẻ độc giả	QLDG_BM 1 QLDG_BM 2	Độc giả có yêu cầu làm thẻ mượn sách sẽ được nhận phiếu đăng ký để điền thông tin vào (QLDG_BM 1), sau đó bộ phận quản lý độc giả tiến hành cấp thẻ và thu lệ phí theo quy định (QLDG_BM 2)
2	Gia hạn thẻ độc giả	Lưu trữ	Gia hạn thẻ theo yêu cầu của độc giả và thời gian quá hạn không được quá 3 tháng. Sau thời gian 3 tháng, những thẻ hết hạn sẽ bị hủy.		
3	Huỷ thẻ độc giả	Lưu trữ	Hủy bỏ các thẻ độc giả đã quá hạn đăng ký 3 tháng.		

QLDG_BM 1:

PHIẾU ĐĂNG KÝ LÀM THẺ MƯỢN SÁCH

Họ và tên: _____ Năm sinh: _____

Địa chỉ thường trú: _____

Nghề nghiệp: _____

Ngày đăng ký: _____

QLDG_BM 2:

THẺ ĐỘC GIẢ

Họ và tên: _____

Trường: _____ Lớp: _____

Địa chỉ: _____

Ngày ___ tháng ___ năm ___

Bộ phận: Quản lý sách. **Mã số: QLS**

stt	Công việc	Loại	Quy định/ Công thức liên quan	Biểu mẫu liên quan	Ghi chú
1.	Nhận sách mới vào kho sách.	Lưu trữ		QLS_BM 1	Khi có sách mới nhập về, bộ phận quản lý sách có trách nhiệm rà xét xem số sách đó đã có hay chưa, nếu chưa thì lập thẻ quản lý sách và định mã số sách mới. Nếu có rồi thì gọi lại thẻ cũ để cập nhật bổ sung số lượng.
2.	Thanh lý sách cũ	Lưu trữ	Các sách hư, không đọc được		
3.	Lập báo cáo các sách cần thanh lý	Kết xuất		QLS_BM 2	
4.	Lập báo cáo sách mượn	Kết xuất		QLS_BM 3	

QLS_BM 1:

THẺ QUẢN LÝ SÁCH

Tên sách: _____

Tập: _____ Số trang: _____

Số lượng: _____ Năm xuất bản: _____

Mã ngôn ngữ: _____ Ngôn ngữ: _____

Mã nhà xuất bản: _____ Nhà xuất bản: _____

Mã phân loại: _____ Phân loại: _____

Mã tác giả: _____ Tác giả: _____

Mã vị trí: ____ Khu: ___ Kệ: ___ Ngăn: ___

QLS_BM 2:

DANH SÁCH CÁC SÁCH CẦN THANH LÝ

stt	Mã sách	Tên sách	Tác giả	Năm sản xuất	Ngày nhập kho	Tình trạng
1						
2						

Ngày lập báo cáo:

Người lập:

QLS_BM 3:

BÁO CÁO THỐNG KÊ SÁCH MƯỢN

Từ ngày _____ Đến ngày _____

stt	Mã sách	Tên sách	Tác giả	Số lượt mượn
1.				
2.				

Ngày lập báo cáo:

Người lập:

b. Xác định yêu cầu chức năng hệ thống và yêu cầu chất lượng

** Cách tiến hành:*

Chuyên viên tin học và nhà chuyên môn cùng đề xuất và cùng xem xét lại các yêu cầu.

** Bước tiến hành:*

Bước 1: Xem xét các yêu cầu chức năng hệ thống cơ bản, thông dụng (yêu cầu phát sinh thêm do thực hiện các công việc trên máy tính): phân quyền, sao lưu, phục hồi, định cấu hình hệ thống, …

Bước 2: Xem xét các yêu cầu chức năng hệ thống chuyên biệt (yêu cầu về các công việc mới, chỉ có thể tiến hành khi thực hiện trên máy tính.

Bước 3: Xem xét các yêu cầu về chất lượng theo từng loại tiêu chuẩn sau:

- Tiến hóa
- Tiện dụng
- Hiệu quả
- Tương thích

Sau đó lập bảng yêu cầu tương ứng theo mẫu sau:

STT	Nội dung	Mô tả chi tiết	Ghi chú
1.			
2.			

Mẫu 3: Bảng yêu cầu chức năng hệ thống.

STT	Nội dung	Tiêu chuẩn	Mô tả chi tiết	Ghi chú
1.				
2.				

Mẫu 4: Bảng yêu cầu về chất lượng.

Ví dụ: Xét phần mềm quản lý thư viện (giả sử phần mềm được xây dựng nhằm phục vụ cho 4 bộ phận là: độc giả, thủ thư, ban giám đốc và quản trị hệ thống).

Bảng yêu cầu chức năng hệ thống:

STT	Nội dung	Mô tả chi tiết	Ghi chú
1	Phân quyền sử dụng	- Người quản trị: được phép sử dụng tất cả các chức năng - Độc giả: chỉ tra cứu sách và đăng ký mượn sách - Ban giám đốc: chỉ tra cứu sách và lập các báo cáo thống kê - Thủ thư: tất cả các chức năng, ngoại trừ chức năng phân quyền, sao lưu và phục hồi dữ liệu	

BẢNG YÊU CẦU VỀ CHẤT LƯỢNG HỆ THỐNG

STT	NỘI DUNG	TIÊU CHUẨN	MÔ TẢ CHI TIẾT	GHI CHÚ
1	Cho phép thay đổi quy định tính tiền phạt	Tiến hóa	Người dùng phần mềm có thể thay đổi đơn giá phạt và biên các mức phạt.	
2	Hình thức tra cứu thật tiện dụng, tự nhiên, trực quan. Dễ sử dụng cho cả những người không chuyên tin học.	Tiện dụng	Hỗ trợ khả năng tra cứu gần đúng, tra cứu theo nội dung,...	
3	Cho phép nhập sách mới từ tập tin Excel có sẵn. Các màn hình có sự nhất quán chung	Tương thích	Có thể nhập trực tiếp danh sách các sách mới có trước trên tập tin Excel với cấu trúc hợp lý.	
4	Tốc độ thực hiện việc cho mượn và tra cứu sách nhanh	Hiệu quả	Tối đa 30 giây cho mỗi phiếu mượn sách. Hỗ trợ thiết bị đọc mã vạch. Tối đa 10 giây phải có kết quả tra cứu.	

1.2.4 Khảo sát một số phần mềm tiêu biểu minh họa cho giai đoạn xác định yêu cầu.

A. Phần mềm hỗ trợ giải bài tập phân số.

Bộ phận: Giáo viên. **Mã số: GV**

STT	Công việc	Loại công việc	Quy định/Công thức liên quan	Biểu mẫu liên quan	Ghi chú
1	Soạn tóm tắt lý thuyết và ví dụ minh họa	Lưu trữ			
2	Soạn đề bài tập	Lưu trữ	GV_QĐ 2	GV_BM 2	
3	Soạn đáp án	Lưu trữ	GV_QĐ 3	GV_BM 3	
4	Chấm điểm	Tính toán	GV_QĐ 4		

STT	Mã số	Tên Quy định/ Công thức	Mô tả chi tiết	Ghi chú
1.	GV_QĐ2	Quy định soạn đề bài tập	• Đề bài được giới hạn chỉ là biểu thức các phép toán trên phân số với tối đa 4 phân số thành phần. • Có 3 mức bài tập: 1. Chỉ gồm 2 phân số và 1 phép toán. 2. Chỉ gồm 3 phân số và 2 phép toán. 3. Hỗn hợp nhiều phân số (tối đa 4 phân số) với nhiều phép toán • Có 4 loại phép toán : + - * /	
2.	GV_QĐ3	Quy định soạn đáp án bài tập (cũng là quy định soạn bài giải của học sinh)	Mỗi bước giải chỉ được phép rút gọn biểu thức bằng các thực hiện phép tính trên 2 phân số. Thứ tự thực hiện phép tính theo quy tắc độ ưu tiên như sau : Ưu tiên 1 : nhân chia cao hơn công trừ. Ưu tiên 2 : bài toán ưu tiên bên phải Riêng đối với bài giải của học sinh cho phép bỏ qua các bước trung gian.	

3.	GV_QĐ4	Quy định chấm điểm	• Có đáp án cuối cùng đúng ➢ Thực hiện hơn hoặc bằng 50% các bước so với đáp án : o Đã rút gọn : 10 o Chưa rút gọn : 8 ➢ Thực hiện dưới 50% các bước so với đáp án : o Đã rút gọn : 9 o Chưa rút gọn : 7 • Có đáp án cuối cùng sai ➢ Thực hiện hơn hoặc bằng 70% các bước so với đáp án : 5 ➢ Thực hiện từ 50% đến dưới 70% các bước so với đáp án : 3 ➢ Thực hiện từ 50% các bước so với đáp án : 0	

GV_BM 2:

Đề bài tập của giáo viên.

Thực hiện các phép tính trên biểu thức các phân số :

<phân số> [phép toán] <phân số> [phép toán] ...

GV_BM 3:

Đáp án của giáo viên (bài giải của học sinh)

Đề bài: _____

Các bước biến đổi tương đương :

Bước 1: ...

Bước 2: ...

Bước 3: ...

Đáp số: ...

Bộ phận: Học sinh. Mã số: HS

stt	**Công việc**	**Loại công việc**	**Quy định liên quan**	**Biểu mẫu liên quan**	**Ghi chú**
1	Chọn bài tập	Tra cứu	GV_QĐ 2	GV_BM 2	
2	Giải bài tập	Lưu trữ	GV_QĐ 3	GV_BM 3	
3	Xem tóm tắt lý thuyết	Kết xuất			
4	Xem đánh giá và đáp án	Kết xuất	GV_QĐ 3 GV_QĐ 4	GV_BM 3	

2. Mô hình hóa yêu cầu hệ thống

Các mô tả yêu cầu trong giai đoạn xác định yêu cầu chỉ mô tả chủ yếu các thông tin liên quan đến việc thực hiện các nghiệp vụ trong thế giới thực chưa và chưa thể hiện rõ nét việc thực hiện các nghiệp vụ này trên máy tính. Mô tả thông qua các văn bản dễ gây ra nhầm lẫn và không trực quan.

Ví dụ: Xét yêu cầu lập hóa đơn bán sách, yêu cầu này chỉ mô tả biểu mẫu về hóa đơn, qui định lập hóa đơn và chưa thể hiện cách thức lập hóa đơn trên máy tính

Mục tiêu của mô hình hóa: Cho phép ta hiểu 1 cách chi tiết hơn về ngữ cảnh vấn đề cần giải quyết một cách trực quan và bản chất nhất (thông tin cốt lõi) yêu cầu.

Kết quả: cho một mô hình mô tả lại toàn bộ hoạt động của hệ thống thực. Mỗi phương pháp phân tích đưa ra một kiểu sơ đồ hay mô hình để xây dựng hệ thống.

Kỹ thuật phân tích là cách tiến hành sao cho thu thập được những yêu cầu của người sử dụng từ đó trình bày lại nhu cầu đó trên mô hình, chi tiết hóa sơ đồ hay mô hình bằng đặc tả chức năng, đặc tả dữ liệu thông qua phân tích gốc nhìn, phân tích đối tượng, phân tích dữ liệu thu thập được ở các bước trên. Trước khi đi vào tìm hiểu các phương pháp biểu diễn bằng mô hình, chúng ta hãy xem qua một số nguyên lý phân tích.

2.1 Các nguyên lý mô hình hóa

a. Mô hình hóa miền thông tin (nguyên lý phân tích 1)

Phải hiểu và biểu diễn được miền thông tin

- Định danh dữ liệu (đối tượng, thực thể)
- Định nghĩa các thuộc tính
- Thiết lập các mối quan hệ giữa các dữ liệu

b. Mô hình hóa chức năng (nguyên lý phân tích 2)

Bản chất của phần mềm là biến đổi thông tin

- Định danh các chức năng (biến đối thông tin)
- Xác định cách thức dữ liệu (thông tin) di chuyển trong hệ thống
- Xác định các tác nhận tạo dữ liệu và tác nhân tiêu thụ dữ liệu

c. Mô hình hóa hành vi (nguyên lý phân tích 3)

Phần mềm (hệ thống) có trạng thái (hành vi)

- Xác định các trạng thái hệ thống

ví dụ: giao diện đồ họa, section trong ứng dụng web

- Xác định các dữ liệu làm thay đổi hành vi hệ thống

ví dụ: bàn phím, chuột, các cổng thông tin…

d. Phân hoạch các mô hình (Nguyên lý phân tích 4)

Làm mịn, phân hoạch và biểu diễn các mô hình ở các mức khác nhau

- Làm mịn các mô hình dữ liệu
- Tạo cây (mô hình) phân rã chức năng
- Biễu diễn hành vi ở các mức chi tiết khác nhau

e. Tìm hiểu vấn đề bản chất (Nguyên lý phân tích 5)

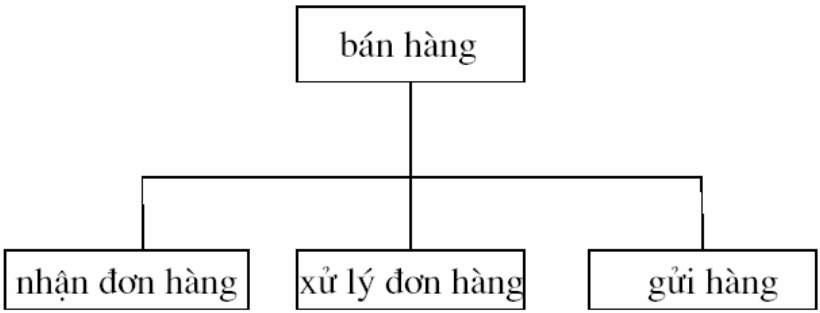
- Nhìn nhận bản chất của yêu cầu
- Không quan tâm đến cách thức cài đặt

2.3 Sơ đồ phân rã chức năng

Sơ đồ phân rã chức năng - Function Decomposition Diagram - FDD: Nêu lên các chức năng thông qua việc mô tả các tính chất của đầu vào và đầu ra

- Xác định phạm vi của hệ thống
- Phân hoạch chức năng
- Tạo nền tảng cho thiết kế kiến trúc hệ thống

Ví dụ: Sơ đồ phân rã chức năng



2.3 Mô hình bản mẫu (protoype)

Khi xác định yêu cầu, nhà phát triển phần mềm dựa trên các ý tưởng hay yêu cầu của khách hàng đưa ra một bản thiết kế sơ bộ một số màn hình giao diện và tiến hành mô phỏng hay giả lập sơ bộ một số chức năng, Có thể xem đây bước cài đặt bản mẫu đầu tiên và chuyển cho người sử dụng. Bản mẫu này chỉ nhằm để mô tả cách thức phần mềm hoạt động cũng như cách người sử dụng tương tác với hệ thống. Nhằn giúp cho người dùng hình dung được diện mạo ban đầu của yêu cầu mà họ đặt ra. Mô hình này cũng cần có sự hỗ trợ giữa kỹ sư phân tích và kỹ sư thiết kế phần mềm phối hợp thực hiện.

Người sử dụng khi xem xét bản mẫu sẽ đưa ra ý kiến đóng góp và phản hồi thông tin đồng ý hay không đồng ý phương án thiết kế của bản mẫu đã đưa ra. Nếu người sử dụng đồng ý với bản mẫu đã đưa thì người phát triển sẽ tiến hành cài đặt thực sự. Ngược lại cả hai phải quay lại giai đoạn xác định yêu cầu. Công việc này được lặp lại liên tục cho đến khi người sử dụng đồng ý với các bản mẫu do nhà phát triển đưa ra.

2.4 Sơ đồ luồng dữ liệu

Sơ đồ luồng dữ liệu - Data flow diagram – DFD

Đây là mô hình cho phép xem toàn bộ sơ đồ luồng dữ liệu bên trong hệ thống. Cách thức dữ liệu được xử lý bên trong hệ thống.Có nhiều mức chi tiết khác nhau. Có nhiều biến thể mở

rộng khác nhau. Xem chi tiết ở chương kế tiếp thiết kế phần mềm. Ngoài ra còn có mô hình thực thể kết hợp được trình bày trong hầu hết các cuốn sách Cơ sở dữ liệu hoặc Thiết kế CSDL.

2.5 Mô hình hướng đối tượng

Phương pháp phân tích hướng đối tượng hình thành giữa thập niên 80 dựa trên ý tưởng lập trình hướng đối tượng. Phương pháp này đã phát triển, hoàn thiện và hiện nay rất phổ dụng. Nó dựa trên một số khái niệm cơ bản sau:

Đối tượng (Object): gồm dữ liệu và thủ tục tác động lên dữ liệu này.

Đóng gói (Encapsulation): Không cho phép tác động trực tiếp lên dữ liệu của đối tượng mà phải thông qua các phương pháp trung gian.

Lớp (Class): Tập hợp các đối tượng có chung một cấu trúc dữ liệu và cùng một phương pháp.

Kế thừa (Heritage): tính chất kế thừa là đặc tính cho phép định nghĩa một lớp mới từ các lớp đã có bằng cách thêm vào đó những dữ liệu mới, các phương pháp mới có thể kế thừa những đặc tính của lớp cũ.

a. Mô hình nắm bắt yều cầu hướng đối tượng bằng UML

Mục đích của hoạt động nắm bắt yêu cầu là xây dựng mô hình hệ thống mà sẽ được xây dựng bằng cách sử dụng các use-case. Các điểm bắt đầu cho hoạt động này khá đa dạng:

- Từ mô hình nghiệp vụ (business model) cho các ứng dụng nghiệp vụ.
- Từ mô hình lĩnh vực (domain model) cho các ứng dụng nhúng (embeded)
- Từ đặc tả yêu cầu của hệ thống nhúng được tạo bởi nhóm khác và hoặc dùng các phương pháp đặc tả khác (thí dụ hướng cấu trúc.

- Từ điểm nào đó nằm giữa các điểm xuất phát trên.

Mô hình use-case:

- Actor: người/ hệ thống ngoài/ thiết bị ngoài tương tác với hệ thống
- Use-case: các chức năng có nghĩa của hệ thống cung cấp cho các actor
  - luồng các sự kiện (flow of events)
  - các yêu cầu đặc biệt của use-case
- Đặc tả kiến trúc
- Các thiết kế mẫu giao diện người dùng

b. Mô hình phân tích hướng đối tượng với UML

Mục đích của hoạt động phân tích yêu cầu là xây dựng mô hình phân tích với các đặc điểm sau:

- Dùng ngôn ngữ của nhà phát triển để miêu tả mô hình
- Thể hiện gốc nhìn từ bên trong hệ thống
- Được cấu trúc từ các lớp phân tích và các package phân tích
- Được dùng chủ yếu cho các nhà phát triển để hiểu cách thức tạo hình dạng hệ thống
- Loại trừ mọi chi tiết dư thừa, không nhất quán
- Phát họa hiện thực các chất năng bên trong hệ thống
- Định nghĩa các dẫn xuất use-case, mỗi dẫn xuất use-case cấp phân tích miêu tả sự phân tích 1 use-case

Mô hình phân tích= hệ thống phân tích

- Các class phân tích: lớp biên, lớp thực thể, lớp điều khiển
- Các dẫn xuất use-case cấp phân tích: các lược đồ lớp phân tích, các lược đồ tương tác, luồng sự kiện, các yêu cầu đặc biệt của use-case
- Các package phân tích
- Đặc tả kiến trúc

Lưu ý: Các mô hình hướng đối tượng cho từng giai đoạn phát triển phần mềm được trình bày ở giáo trình khác. Xem chi tiết cụ thể ở giáo trình môn Phân tích thiết kế hướng đối tượng với UML.

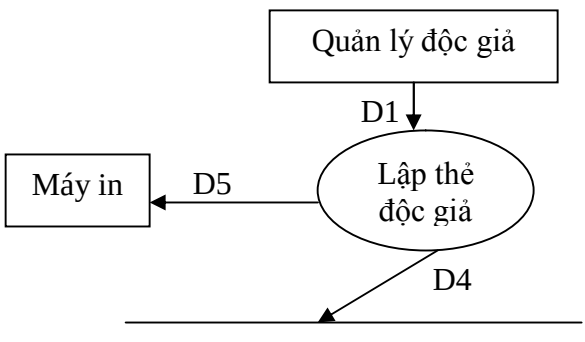
2. 6 Ví dụ minh họa từ yêu cầu sang mô hình hóa

Ví dụ 1: Xét phần mềm quản lý thư viện với 4 yêu cầu

- Lập thẻ độc giả
- Nhận sách
- Cho mượn sách
- Trả sách

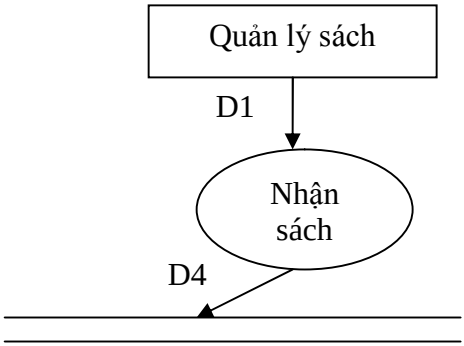
Giai đoạn 2 : Mô hình hóa yêu cầu

- Sơ đồ luồng dữ liệu cho công việc lập thẻ độc giả



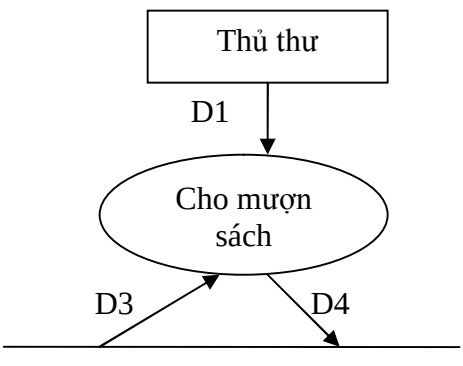
D1: Thông tin về thẻ độc giả cần nhập
D4: Thông tin về thẻ độc giả cần lưu trữ trên bộ nhớ phụ
D5: Thông tin trên thẻ độc giả (trong thế giới thực)
Xử lý thẻ độc giả: Kiểm tra tính hợp lệ của thẻ trước ghi nhận và in

- Sơ đồ luồng dữ liệu cho công việc nhận sách



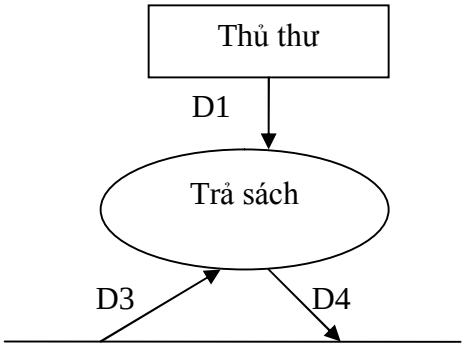
D1: Thông tin về thẻ sách cần nhập
D4: Thông tin về sách cần lưu trữ trên bộ nhớ phụ
Xử lý nhập sách: Kiểm tra tính hớp lệ của sách trước khi ghi nhận trên bộ nhớ phụ

- Sơ đồ luồng dữ liệu cho công việc cho mượn sách



D1: Thông tin về độc giả và sách muốn mượn
D3: Thông tin được sử dụng cho việc kiểm tra qui định mượn sách
D4: Thông tin về việc mượn sách
Xử lý cho mượn sách: Kiểm tra tính hợp lệ của việc mượn sách ghi nhận trên bộ nhớ phụ

- Sơ đồ luồng dữ liệu cho công việc trả sách



D1: Thông tin về độc giả và sách trả
D3: Thông tin sử dụng cho việc kiểm tra qui định trả sách
D4: Thông tin về việc trả sách
Xử lý trả sách: Kiểm tra tính hợp lệ của việc trả sách ghi nhận trên bộ nhớ.

Chương 3: THIẾT KẾ PHẦN MỀM

1. Tổng quan về thiết kế

Trong thiết kế, chúng ta định hình hệ thống và tìm dạng thức của nó (kể cả kiến trúc) mà đáp ứng được mọi yêu cầu, cả yêu cầu phi chức năng và các ràng buộc khác - được đặt ra cho hệ thống đó. Một đầu vào cơ bản cho thiết kế là kết quả thu được từ phần tích, đó là mô hình phân tích. Xét một cách chi tiết mục đích của thiết kế là:

- Thu được sự hiểu biết sâu về các yêu cầu phi chức năng và các ràng buộc có liên quan tới ngôn ngữ lập trình, sử dụng lại thành phần, các hệ điều hành, các công nghệ phân tán, các công nghệ cơ sở dữ liệu, các công nghệ giao diện người dùng, các công nghệ quản lý các giao dịch.
- Tạo ra một đầu vào thích hợp và xuất phát điểm cho các hoạt động cài đặt tiếp theo sau bằng cách nắm bắt các yêu cầu về mỗi hệ thống cụ thể, các giao diện, và các lớp.
- Có khả năng phân rã việc cài đặt thành các mẩu nhỏ dễ quản lý hơn được nhiều đội phát triển khác nhau xử lý và có thể tiến hành đồng thời. Điều này sẽ có ích trong các trường hợp khi mà không thể tiến hành sự phân rã giữa các kết quả thu được từ nắm bắt các yêu cầu hoặc phân tích.
- Nắm bắt sớm các giao diện chủ yếu giữa các hệ thống con trong vòng đời của phần mềm. Điều này sẽ có ích khi chúng ta suy luận về kiến trúc và khi chúng ta sử dụng các giao diện như những công cụ đồng bộ các đội phát triển khác nhau

- Trực quan hóa và suy luận thiết kế bằng cách sử dụng một hệ thống các ký pháp chung.
- Tạo ra một sự trừu tượng hóa liên tục của việc cài đặt của hệ thống, tức là cài đặt sự làm mịn dần thiết kế bằng cách đắp "thịt" vào khung xương nhưng không thay đổi cấu trúc của nó.

Mục tiêu của phần này là giới thiệu một số phương pháp và kỹ thuật chính trong thiết kế, đối với việc triển khai một hệ thống thành nhiều hệ thống con và hệ thống con thành nhiều thành phần (components), và quản lý những vấn đề liên quan đến cấu trúc nội tại của những thành phần hệ thống. Đầu tiên chúng ta xem qua vài kỹ thuật thiết kế. Kế đến chúng ta sẽ xét qua một vài kỹ thuất thiết kế và phương pháp nền tảng một cách chi tiết và một số ví dụ minh họa. Thêm vào đó, chúng ta bàn qua những khía cạnh thiết kế như thiết kế giao diện người dùng và mô đun hóa.

1.1 Kỹ thuật thiết kế

- Thiết kế đặc tả đi đến kỹ thuật cốt lõi của tiến trình của công nghệ phần mềm.
- Thiết kế đặc tả được cung cấp xem xét những mô hình của tiến trình phần mềm được sử dụng.
- Thiết kế phần mềm là bước đầu tiên trong ba hoạt động kỹ thuật - thiết kế, phát sinh mã nguồn, và thử nghiệm – đó là những yêu cầu trong xây dựng và phát triển phần mềm.

Một trong những điểm mấu chốt chính đối với độ phức tạp của hệ thống phần mềm là sự trừu tượng. Có hai phương pháp chính: thiết kế Top-down và thiết kế bottom-up

1.1.1 Thiết kế trên xuống (Top-down)

-Thiết kế bắt đầu với việc phân tích những định nghĩa yêu cầu và không nên xem xét việc thực hiện chi tiết đầu tiên.

- Một dự án được triển khai thành những dự án nhỏ, thủ tục này phải được lặp lại cho đến khi những nhiệm vụ con trở nên đơn giản sao cho một thuật toán được tính toán và giải quyết.

1.1.2 Thiết kế từ dưới lên (Bottom–up)

Ý tưởng nền tảng: Hiểu được phần cứng và tầng trên của nó như một cơ chế trừu tượng.

Kỹ thuật: Thiết kế từ dưới lên bắt đầu được cho bởi máy cụ thể và liên tiếp phát triển một máy trừu tượng sau khi những máy khác được thêm vào những thuộc tính cần thiết cho đến khi một máy đã đạt được kết quả mà cung cấp những chức năng người dùng yêu cầu.

1.1.3 Thiết kế hệ thống

Trong hệ thống lớn, tiến trình thiết kế bao gồm một yếu tố thiết kế hệ thống mà chức năng được phân chia thành những chức năng phần mềm và phần cứng.

Những thuận lợi của chức năng thực hiện trong phần cứng là thành phần phần cứng phân phối thực hiện tốt hơn đơn vị phần cứng. Nút thắt của hệ thống được xác định và thay thế bởi thành phần của phần cứng, như thế việc tối ưu phần mềm là hết sức tốn kém.

Cung cấp tốc độ phần cứng có nghĩa là thiết kế phần mềm có thể được cấu trúc cho khả năng thích ứng và khả năng xem xét thực thi cả chức năng.

1.1.4 Thiết kế bản mẫu (prototype)

Thiết kế bản mẫu nghĩa là đưa ra các màn hình giao diện sơ bộ, hay các bản thiết kế phác thảo nháp cho người dùng tham khảo trước khi đi vào thiết kế chi tiết, hay chức năng cụ thể. Các bản thiết kế này được soạn thảo dưới dạng sưu liệu hoặc một số phần mềm có khả năng thiết kế nhanh giao diện, các kỹ

sư thiết kế có thể sử dùng một số phần mềm chuyên dụng để soạn thảo nhanh như MS Visual Basic, Visual C++, MS Visual Studio … với trang web thì có thể dùng Front Page, MS Visual Interdev chỉ với những đoạn chương trình đơn giản được cài đặt. Đây cũng có thể coi là bước đệm cơ bản trước khi đi vào cài đặt chi tiết cho từng chương trình con hay môđun con v.v.

1.1.5 Phân rã thiết kế

Tiến trình thiết kế không chỉ ảnh hưởng đến phương pháp thiết kế mà còn ảnh hướng đến tiêu chuẩn được sử dụng để phân rã hệ thống.

Phần lớn những yếu tố cơ bản của phân rã được đề ra.

Phương pháp phân loại phân rã

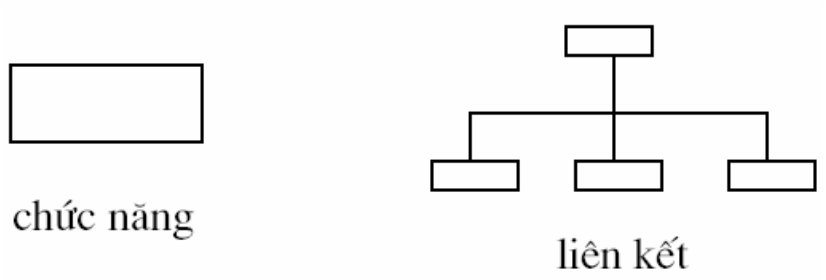
1.1.5.1 Phân rã hướng chức năng

- Khía cạnh của hệ thống hướng chức năng tạo nên cốt lõi của thiết kế

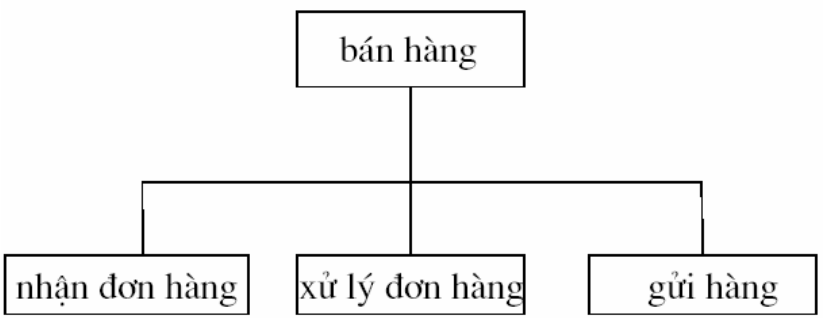
- Dựa trên những yêu cầu chức năng chứa trong những định nghĩa yêu cầu, phân rã hướng đến tác nhiệm của toàn bộ hệ thống được tổ chức

Sơ đồ phân rã chức năng - Function Decomposition Diagram - FDD: Nêu lên các chức năng thông qua việc mô tả các tính chất của đầu vào và đầu ra

- Xác định phạm vi của hệ thống
- Phân hoạch chức năng
- Tạo nền tảng cho thiết kế kiến trúc hệ thống



Ví dụ: **Sơ đồ phân rã chức năng**



1.1.5.2 Phân rã hướng dữ liệu

Tiến trình thiết kế tập trung trên khía cạnh hệ thống hướng đến dữ liệu. Chiến lược thiết kế hướng đến chính dữ liệu đực thực hiện. Phân rã những bộ phận hệ thống từ việc phân tích dữ liệu

1. Sơ đồ luồng dữ liệu

Sơ đồ luồng dữ liệu - Data flow diagram - DFD

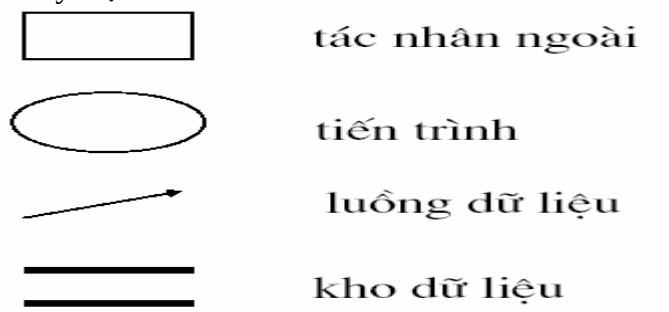
Cho phép xem toàn bộ sơ đồ luồng dữ liệu bên trong hệ thống. Cách thức dữ liệu được xử lý bên trong hệ thống.Có nhiều mức chi tiết khác nhau. Có nhiều biến thể mở rộng khác nhau

a. Khái niệm và ký hiệu

Tác nhân ngoài: đối tượng bên ngoài hệ thống, nguồn phát sinh hay thu nhận dữ liệu

Tiến trình: Thao tác đối với thông tin hay khối dữ liệu
Luồng dữ liệu: luồng thông tin di chuyển trong hệ thống
Kho dữ liệu:nơi lưu trữ dữ liệu

Các ký hiệu:



b. Các nguyên tắc và bước xây dựng mô hình DFD

Các bước xây dựng DFD:

- Phân rã chức năng hệ thống
- Liệt kê các tác nhân, các khoản mục dữ liệu
- Vẽ DFD cho các mức

Nguyên tắc:

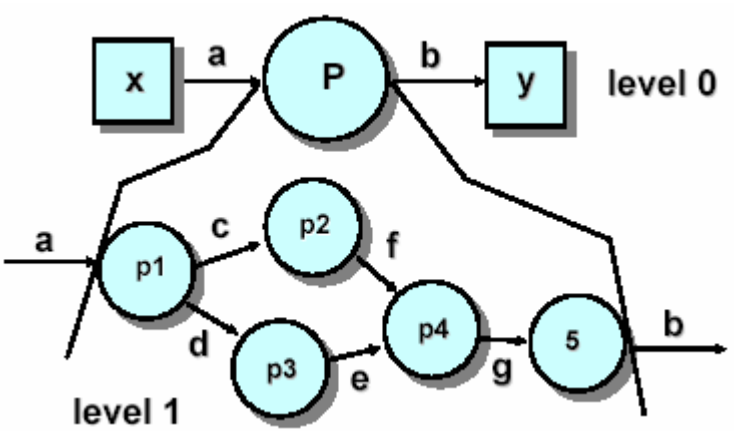
- Các tiến trình phải có luồng vào luồng ra
- Không có luồng dữ liệu trực tiếp giữa các tác nhân với tác nhân và kho dữ liệu
- Luồng dữ liệu không quay lại nơi xuất phát
- Bắt đầu bằng DFD mức 0, liệt kê các tác nhân ngoài ở mức 0
- Các mức(cấp) sơ đồ:

- mức 0: Toàn bộ phần mềm là khối xử lý
- mức 1: Sơ đồ mức 0 có thể phân rã thành nhiều sơ đồ mức 1, các sơ đồ mức 1này phải đảm bảo thể hiện đầy đủ ý nghĩa sơ đồ mức 0 (tác nhân, thiết bị, luồng dữ liệu, xử lý, bộ nhớ phụ)
- mức 2: Mỗi sơ đồ mức 1 có thể phân rã thành nhiều sơ đồ mức 2 tương ứng như việc phân rã của sơ đồ mức 0
- …

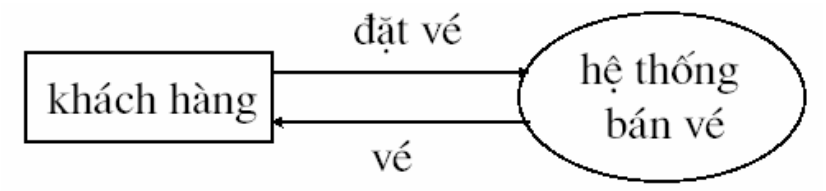
Trình bày sơ đồ: Trong mỗi cấp có 2 hình thức trình bày sơ đồ

- Dạng tổng hợp : Chỉ có một khối xử lý chung, tất cả các luồng dữ liệu chỉ tập trung liên quan đến khối xử lý chung này
- Dạng chi tiết: Bao gồm nhiều khối xử lý với luồng dữ liệu riêng biệt cho từng khối xử lý

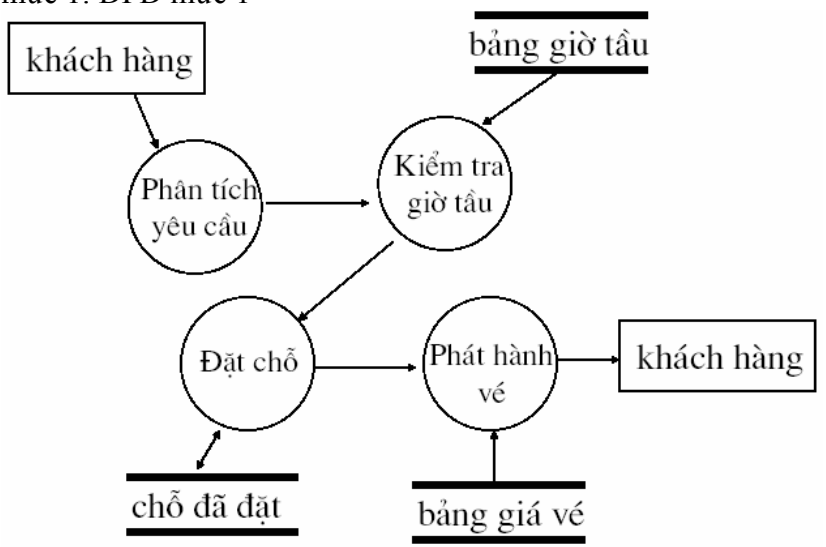
Ví dụ: biểu diễn các mức của DFD



Ví dụ DFD hệ thống bán vé mức 0:



mức 1: DFD mức 1



2. Các hướng tiếp cận lập sơ đồ luồng dữ liệu

- Có nhiều hướng tiếp cận để tạo lập các sơ đồ luồng dữ liệu. Giáo trình này giới hạn xem xét 3 cách tiếp cận chính
  + Hướng tiếp cận từ trên xuống dưới (topdown)
  + Hướng tiếp cận từ dưới lên trên (bottomup)
  + Hướng tiếp cận phối hợp

- **Tiếp cận từ trên xuống:**

Quá trình thực hiện theo hướng tiếp cần này như sau:

- Lập sơ đồ luồng dữ liệu cấp 0 (xem xét tất cả các luồng dữ liệu nhập xuất, tất cả các yêu cầu xử lý của phần mềm
- Phân rã sơ đồ luồng dữ liệu cấp 0 thành nhiều sơ đồ luồng dữ liệu cấp 1. Có 2 cách phân rã:

  + Phân rã các xử lý của phần mềm thành nhiều xử lý con và quyết định các luồng dữ liệu tương ứng trên các xử lý con này.

  + Phân rã các luồng dữ liệu nhập xuất thành nhiều luồng dữ liệu con và quyết định các xử lý tương ứng với các luồng dữ liệu con này.

- Quá trình kết thúc khi đạt đến các sơ đồ không thể tiếp tục phân rã được (sơ đồ lá). Thông thường đây là sơ đồ tương ứng với công việc cụ thể của một nhà chuyên môn trong thế giới thực.

Đánh giá

- Tiếp cận này thích hợp với các phần mềm có số lượng người dùng, số lượng các yêu cầu ít (nếu ngược lại sơ đồ cấp 0 sẽ rất phức tạp và khó lập chính xác).
- Tiếp cận này đặc biệt thích hợp với các loại phần mềm mà vì lý do nào đó các hệ thống yêu cầu chưa được xác định rõ ngay từ đầu (ví dụ các phần mềm hệ thống).
- Thông thường cách tiếp cận này ít đượcc sử dụng.

- Hướng tiếp cận từ dưới lên (bottomup)

Quá trình thực hiện theo hướng tiếp cận này như sau

- Lập sơ đồ luồng dữ liệu ở mức cao nhất. Các sơ đồ này sẽ không được tiến hành phân rã thành các sơ đồ có cấp lớn hơn (thông thường đây là sơ đồ ứng với một công việc cụ thể của một người dùng nào đó trong thế giới thực)

  + Tích hợp các sơ đồ này để tạo lập các sơ đồ có cấp nhỏ hơn (thông thường các sơ đồ được chọn tích hợp

theo một tiêu chí cụ thể: cùng một người sử dụng, cùng một loại yêu cầu, v.v). Có 2 cách tích hợp:

+ Tích hợp các xử lý của các sơ đồ cấp k vào sơ đồ cấp k-1 và giữ nguyên các luồng dữ liệu của các sơ đồ cấp k

+ Tích hợp đồng thời các xử lý và các luồng dữ liệu của các sơ đồ cấp k để tạo lập sơ đồ cấp k-1.

- Quá trình kết thúc khi đạt đến các sơ đồ cấp 0

Đánh giá

- Tiếp cận này rất thích hợp với các phần mềm có hệ thống yêu cầu chi tiết, cụ thể và có qui mô yêu cầu (số lượng người dùng, số lượng yêu cầu) thuộc mức trung bình (các đồ án môn học
- Tiếp cận này sẽ khó khăn nếu qui mô yêu cầu lớn và chưa thật rõ ràng chi tiết
- Cách tiếp cận này sẽ được sử dụng trong giáo trình với các đồ án môn học và các ví dụ minh họa

▪ **Hướng tiếp cận phối hợp:**

Quá trình thực hiện theo hướng tiếp cận này như sau:

- Lập sơ đồ luồng dữ liệu cấp k theo một tiêu chí xác định (sơ đồ cho từng người dùng, sơ đồ cho một bộ phận, sơ đồ cho một loại yêu cầu, v.v)
- Phân rã sơ đồ cấp k thành nhiều sơ đồ cấp k+1 tiếp tục cho đến khi đạt được các sơ đồ lá
- Tích hợp các sơ đồ cấp k thành các sơ đồ cấp k-1 tiếp tục cho đến khi đạt được sơ đồ cấp 0

Đánh giá

- Tiếp cận này thích hợp cho các phần mềm có qui mô yêu cầu lớn, phức tạp
- Tiếp cận này được sử dụng rất thường xuyên trong thực tế.

3. Lập sơ đồ luồng dữ liệu cho từng công việc

- Do các giới hạn đã nêu phía trên việc lập các sơ đồ luồng dữ liệu toàn bộ phần mềm chỉ qui về lập sơ đồ luồng dữ liệu cho từng công việc (sau đó chỉ thực hiện đơn giản một bước tích hợp để có sơ đồ cấp 0)
- Quá trình lập sơ đồ luồng dữ liệu cho một công việc được tiến hành qua các bước như sau
    - Bước 1: Xác định dữ liệu nhập
    - Bước 2: Xác định dữ liệu xuất
    - Bước 3: Mô tả xử lý
- **Bước 1: Xác định dữ liệu nhập**

- Dữ liệu nhập từ người dùng sử dụng đươc xác định dựa vào biểu mẫu có liên quan với các lưu ý sau:

  + Không nhập vào các dữ liệu có thể tính toán được dựa trên qui định hay công thức đã có.

  + Không nhập vào các dữ liệu đã được lưu trữ trước đó (qua một công việc khác).

- Dữ liệu nhập từ thiết bị nhập (khác bàn phím) chỉ được xem xét khi có yêu cầu đặc biệt trong một số ứng dụng đặc biệt (hệ thống thời gian thực, hệ thống bản đồ, nhập thông qua sử dụng điện thoại tổng đài điện thoại trong quản lý khách sạn, v.v).

- Dữ liệu nhập (đọc) từ bộ nhớ phụ được xác định dựa trên các qui định công thức liên quan với một số lưu ý:

  + Chỉ đọc dữ liệu thật sự cần thiết cho việc thực hiện xử lý tương ứng (thông tin nhập chưa đủ để xử lý).

  + Để cải tiến chất lượng phần mềm(đặc biệt tính tiến hóa) có thể đọc thêm các tham số phục vụ cho việc xử lý từ bộ nhớ phụ (bảng qui định đơn giá phạt khi trả sách trễ hạn, bảng định mức và đơn giá tiền điện, v,v). Tuy nhiên trong giai đoạn này chỉ nên tập trung vào tính đúng đắn (các chất lượng khác sẽ được xem xét chi tiết trong giai đoạn thiết kế).

- **Bước 2: Xác định dữ liệu xuất**

- Dữ liệu xuất cho người dùng được xác định dựa trên biểu mẫu liên quan với một số lưu ý như sau

  +Các thông báo về việc xử lý có thực hiện được hay không là luôn luôn phải có và không cần thiết thể hiện trên sơ đồ (thông báo việc mượn sách là không hợp lệ, thông báo lỗi khi tính điểm trung bình mà có môn chưa có điểm, v.v)

  + Để tăng tính tiện dụng, trong tất cả các xử lý đều phải xuất cho người dùng nhiều thông tin (kể cả xử lý lưu trữ, xử lý tính toán). **Tuy nhiên vấn đề này chỉ xem xét và thực hiện trong các giai đoạn sau, nếu chú ý quá sớm đến vấn đề này sẽ làm phức tạp sơ đồ và dễ phạm các sai lầm trong tính đúng đắn.**

- Dữ liệu xuất ra thiết bị xuất (khác màn hình) thông thường là máy in, để tăng tính tiện dụng có thể tuân theo nguyên tắc sau "Tất cả dữ liệu xuất ra màn hình đều cho phép người dùng xuất ra máy in (có thể với cách trình bày khác). Tuy nhiên vấn đề này cũng có thể dời lại xem xét chi tiết trong giai đoạn thiết kế. Các loại thiết bị xuất khác chỉ có trong các loại ứng dụng đặc biệt hoặc do yêu cầu tính tương thích.

- Dữ liệu xuất (ghi) vào bộ nhớ phụ được xác định dựa trên biểu mẫu liên quan với một số lưu ý như sau:

  + Ghi các dữ liệu kết quả mới tạo lập hoặc các dữ liệu đã có nhưng bị thay đổi trong quá trình thực hiện xử lý.

  + Để tăng tính hiệu quả có thể ghi các thông tin bổ sung có liên quan đến các yêu cầu khác. Tuy nhiên tốt nhất vấn đền này được xem xét chi tiết trong giai đoạn thiết kế.

- **Bước 3: Mô tả xử lý**

Mô tả quá trình sử dụng dữ liệu nhập D1, D2, D3 để tạo ra các dữ liệu xuất D4, D5, D6 với các lưu ý sau:

- Chỉ mô tả xử lý mà không cần lưu ý đến cách thực hiện nhập xuất (hình thức nhập, tổ chức lưu trữ trên bộ nhớ phụ, câu lệnh cụ thể để đọc, ghi).
- Mô tả chi tiết cách sử dụng dữ liệu nhập để tạo dữ liệu xuất (mô tả càng chi tiết thì việc thiết kế xử lý càng dễ dàng.
- Chỉ chú trọng đến tính đúng đắn mà không nên xem xét quá sớm các yêu cầu chất lượng khác.

Mô tả chính xác thứ tự nhập và xuất (trong một vài trường hợp có thể xuất trước và sau đó mới nhập).

2. Mô hình thực thể quan hệ (Entity – Relation Diagram)

a.Các khái niệm và ký hiệu

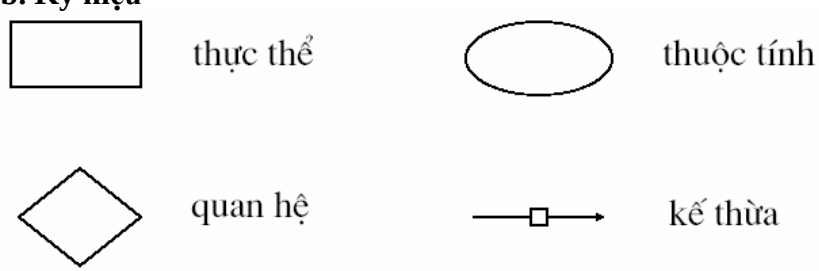
Thực thể là đối tượng thế giới thực mà chúng ta muốn xử lý, có thể là đối tượng thực hay trừu tượng

Thuộc tính: đặc điểm của thực thể

Quan hệ: là mối liên hệ giữa các thực thể, là thông tin cần lưu trừu/ xử lý

Kế thừa: là quan hệ kế thừa giữa các thực thể

b. Ký hiệu



Với giáo trình này chỉ dừng lại giới thiệu khái niệm, mô hình được học ở các giáo trình Phân tích Thiết kế hệ thống thông tin

1.1.5.3 Phân rã hướng đối tượng

Khía cạnh hệ thống hướng đối tượng cung cấp tập trung chủ yếu của thiết kế.

Hệ thống phần mềm được xem xét như tập hợp các đối tượng thông tin với nhau. Mỗi đối tượng có cấu trúc dữ liệu mà không được nhìn thấy từ bên ngoài và thao tác của chúng có thể được thực hiện trên cấu trúc này.

Những điểm cơ bản của phân rã hướng chính nó đến tính đồng nhất giữa dữ liệu và thao tác và dựa trên sự che dấu thống tin và dẫn xuất kế thừa.

1.2. Thiết kế giao diện người dùng

Thiết kế giao diện người dùng là một tác nhiệm trong giai đoạn thiết kế. Thiết kế giao diện được hỗ trợ một phần trong thiết kế dạng mô hình bản mẫu (prototype) ở giaiđoạn xác định nhằm làm sáng tỏ các yêu cầu từ người dùng, xác định đúng yêu cầu người dùng, cũng như thõa mãn các đòi hỏi về mặt thẩm mỹ, giao diện đẹp cho khách hàng. Nếu khách hàng đã đồng ý với bản mẫu đã đưa ra trong giai đoạn xác định yêu cầu thì kỹ sư thiết kế chỉ việc phát triển và hoàn chỉnh thêm giao diện để đảm bảo tính tiện dụng, đảm bảo chính xác yêu cầu người dùng. Nếu không, người thiết kế phải sáng tạo thêm theo một số tiêu chí về thẩm mỹ, tiện dụng, đầy đủ yêu cầu thông tin:

Chế độ (Modes):

Chế độ chương trình là trường hợp mà người dùng chỉ có thể thực hiện ở một số thao tác giới hạn.Kỹ thuật cửa sổ cung cấp dịch vụ có giá trị của chế độ chương trình.

Cửa sổ trợ giúp, người dùng có thể thực hiện vài thao tác con tương ứng trong những cửa sổ khác nhau thể hiển bởi những chế độ chương trình khác nhau.

Thực đơn (Menu):

Pop-up menu: thiết kế hiệu quả bởi chúng có thể xuất hiện bất cứ vị trí nào và ít đòi hỏi di chuyển chuột (mouse).

Pull-down menu: cho phép cấu trúc tốt hơn việc mở rộng tập lệnh và dễ dàng sử dụng.

Người dùng chọn vào thực đơn bằng chuột để hiển thị tất cả lệnh thao tác trên menu và có thể chọn lệnh giống như sử dụng chuột click vào menu. Chúng ta có thể phân loại menu theo tập lệnh thao tác, tập lệnh thao tác với tham số, tập lệnh chuyển đối chế độ người dùng.

1.3.Cửa sổ hội thoại (dialog window):

Đảm bảo tính đồng nhất trong giao diện người dùng, tránh những giải thích dài dòng nên ngắn gọn cô động như cách đặt nhãn Label, Checkbox, Button, List box.

Màu sắc (Color):

Màu sắc chủ yếu chỉ dùng ở những nơi cần diễn đạt những yêu cầu nào đó, hay muốn nhấn mạnh ý nghĩa nào đó, hoặc dấu hiệu cảnh báo nguy hiểm, cùng đừng hóa tô điểm qúa cho giao diện. Ví dụ màu chữ đen trên nên trắng thường dễ đọc nhất cho khả năng làm việc hàng ngày, còn màu chữ trắng trên nền xanh thì khó đọc …

Âm Thanh (Sound):

Cách tốt nhất tập trung sự chú ý của người dùng. Ứng dụng phù hợp trong các tình huống xử lý lỗi, sự kiện không chắc chắn, tạm thời. Tạo những âm thanh khác nhau với những sự kiện khác nhau, tránh dùng âm thanh gây ồn.

Tính kiên định:

Menu lệnh với những chức năng giống nhau nên vị trí giống nhau thậm chí ở những chương trình khác nhau. Phím nóng trên menu lệnh nên cố định. Nút lệnh với những chức năng tương tự giống nhãn và vị trí liên hệ như nhau trong những cửa sổ hội thoại.

1.4 Thiết kế hướng chức năng

Thiết kế hướng chức năng có nghĩa là tập trung trên thuật toán để giải quyết vấn đề.

Hãy tưởng tượng một thuật toán như một hàm tính toán mà tính kết quả từ những tham số cơ bản được cho. Tại thời điểm bắt đầu giai đoạn thiết kế, thuật toán như là hộp đen mà nội dung thì không được biết. Những tác nhiệm càng khó để giải quyết là thuật toán giải quyết của nó. Như vậy, rõ ràng thực hiện mô đun hóa để phân rã những tác nhiệm thành tác nhiệm con độc lập nhau, nhờ đó những thuật toán cho những giải quyết của tác nhiệm con được xem như là những hộp đen. Kết quả chung của những giải pháp trở thành mạng những thuật toán con gộp lại.

1.5.Thiết kế hướng đối tượng

Thiết kế hướng đối tượng là tổ chức xoay quanh những đối tượng và mối liên hệ giữa chúng

Thiết kế lớp đối tượng: mô tả các lớp đối tượng (thuộc tính, hành động

Thiết kế giao diện: Mô tả giao diện của lớp đối tượng trong từng trách nhiệm của chúng

Thiết kế dữ liệu: Mô tả cách thức tổ chức lưu trữ các đối tượng trên bộ nhớ phụ (chỉ có khi không sử dụng cơ sở dữ liệu hướng đối tượng)

Khả năng dùng lại đóng vai trò quan trong trong lập trình hướng đối tượng đối với chuên viên tin học (phải thực hiện nhiều phần mèm). Với tiếp cận mới việc tái sử dụng sẽ rất dễ dàng, nhanh chóng và tốn ít chi phí nhất có thể có (các phần mềm trong cùng lớp phần mềm bao gồm các đối tượng tương tự như nhau, cách xây dựng đối tượng tương tự như nhau cho các phần mềm khác nhau).

3. Kiến trúc phần mềm

Kiến trúc phần mềm bao gồm các thành phần cơ bản: thành phần giao diện, thành phần xử lý, phần dữ liệu. Khi thiết kế một phần mềm cụ thể, người kỹ sư tin học phải chọn lựa và ra quyết định về các "vật liệu" được dùng trong các thành phần. Sau khi đã quyết định xong, kết quả sẽ được mô tả lại hay đặc tả dưới dạng các bản vẽ phần mềm, dưới dạng sưu liệu.

Kết quả của thiết kế là các mô hình phần mềm. Mô hình cung cấp các thông tin chi tiết về 3 thành phần

- Thành phần giao diện
- Thành phần xử lý
- Thành phần dữ liệu

Thông tin về các thành phần giao diện bao gồm các thông tin sau:

- Nội dung và hình thức trình bày các màn hình giao tiếp của phần mềm. Ý niệm về màn hình giao tiếp sẽ được trình bày chi tiết trong phần thiết kế giao diện.
- Hệ thống các thao tác mà người dùng có thể thực hiện trên màn hình giao tiếp và xử lý tương ứng của phần mềm. Các ý niệm về thao tác và xử lý trên màn hình giao tiếp sẽ được trình bày chi tiết trong phần thiết kế giao diện.

Thông tin về thành phần xử lý bao gồm các thông tin sau:

- Hệ thống các kiểu dữ liệu được sử dụng trong phần mềm. Các kiểu dữ liệu này được mô tả cách tổ chức lưu trữ dữ liệu trong bộ nhớ chính của phần mềm
- Hệ thống các hàm được sử dụng trong phần mềm. Các hàm này sẽ thể hiện tương ứng việc thực hiện một công việc nào đó của thế giới thực trên máy tính (kiểm tra tính hợp lệ việc cho mượn sách, ghi vào sổ việc cho mượn sách…v.v)

Thông tin về các thành phần dữ liệu bao gồm các thông tin liên quan đến cách thức tổ chức lưu trữ các dữ liệu (nội dung của việc ghi chép vào sổ sách trong thế giới thực) trên bộ nhớ phụ.

- Dạng lưu trữ được sử dụng của phần mềm. Ý niệm về dạng lưu trữ (tập tin, cơ sở dữ liệu,..v.v) sẽ được trình bày chi tiết trong phần thiết kế dữ liệu

Hệ thống các thành phần lưu trữ cùng với quan hệ giữa chúng. Ý niệm về thành phần lưu trữ cùng với quan hệ giữa các thành phần này cũng sẽ được trình bày chi tiết trong phần thiết kế dữ liệu

4. Phương pháp thiết kế phần mềm

Tùy thuộc vào qui trình được chọn khi thực hiện phần mềm, việc thiết kế có thể được tiến hành theo 2 phương pháp chính:

- Phương pháp trực tiếp
- Phương pháp gián tiếp

Phương pháp trực tiếp được áp dụng khi thực hiện phần mềm không thông qua giai đoạn phân tích. Với phương pháp này việc thiết kế sẽ nhận kết quả chuyển giao trực tiếp từ giai đoạn xác định yêu cầu. Mô hình phần mềm sẽ được xây dựng trực tiếp từ các yêu cầu. Cách tiếp cận này sẽ rất khó khăn cho người thực hiện với các phần mềm có qui mô lớn (nhiều yêu cầu, yêu cầu phức tạp. v.v).

Với phương pháp trực tiếp, thiết kế phần mềm là quá trình cho phép chuyển đổi từ các yêu cầu (kết quả giai đoạn xác định yêu cầu) đến mô hình phần mềm tương ứng. Mục tiêu chính của việc thiết kế là mô tả các thành phần của phần mềm (thành phần giao diện, thành phần xử lý, thành phần dữ liệu) tương ứng với các yêu cầu của phần mềm (yêu cầu chức năng nghiệp vụ, yêu cầu chức năng hệ thống, yêu cầu phi chức năng.

Phương pháp gián tiếp được áp dụng với các qui trình có giai đoạn phân tích. Với phương pháp này việc thiết kế sẽ chỉ nhận một phần các kết quả chuyển giao trực tiếp từ giai đoạn xác định yêu cầu, phần chính yếu sẽ được nhận gián tiếp qua giai đoạn phân tích.

Mô hình phần mềm sẽ được xây dưng tương ứng theo các mô hình trong giai đoạn phân tích. Cách tiếp cận này sẽ rất thuận lợi trong đa số trường hợp với các phần mềm qui mô lớn.

Với phương pháp gián tiếp, thiết kế phần mềm là quá trình cho phép chuyển từ mô hình thế giới thực (kết quả giai đoạn phân tích) đến mô hình phần mềm tương ứng. Mục tiêu chính của việc thiết kế là mô tả các thành phần của phần mềm (thành phần giao diện, thành phần xử lý, thành phần dữ liệu) tương ứng với các mô hình của thế giới thực (mô hình xử lý, mô hình dữ liệu).

5. Ví dụ minh họa

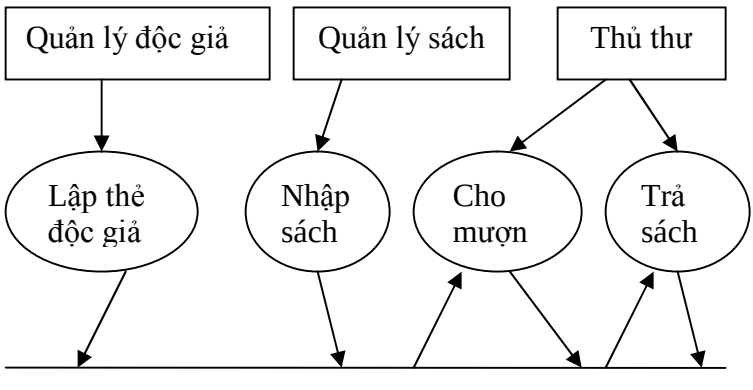
Các ví dụ sau đây chỉ nhằm **minh họa** quá trình thiết kế phần mềm sau khi thực hiện giai đoạn mô hình hóa yêu cầu, các kết quả chỉ chú trọng chủ yếu **tính đúng đắn** và bỏ qua các yêu cầu chất lượng khác (tiến hóa, hiệu quả, tiện dụng). Kết quả thực tế khi xem xét đầy đủ các yêu cầu chất lượng là quá phức tạp và không thích hợp cho việc min họa

Ví dụ 1: Xét phần mềm quản lý thư viện với 4 yêu cầu

- 1. Lập thẻ đọc giả

- 2. Nhận sách
- 3. Cho mượn sách
- 4. Trả sách

a. Mô hình hóa các yêu cầu



b.Thiết kế phần mềm

Hệ thống các màn hình giao diện

- Màn hình chính:

- Nộidung:
  + Thông tin về thư viện
  + Thông tin về các độc giả trong thư viện
  + Thông tin về các sách trong thư viện
- Thao tác người dùng
  + Tra cứu và chọn độc giả
  + Tra cứu và chọn sách

- Màn hình Lập thẻ

- Nộidung:
  + Thông tin về thẻ độc giả
- Thao tác người dùng
  + Nhập thông tin về thẻ
  + Yêu cầu lập thẻ

- Màn hình Cho mượn sách:

- Nộidung:
    + Thông tin về thẻ độc giả
    + Ngày mượn sách
    + Danh sách các sách muốn mượn
- Thao tác người dùng
    + Nhập thông tin về việc cho mượn sách
    + Yêu cầu cho mượn sách

- Màn hình Nhận sách:
- Nộidung:
    + Ngày nhận sách
    + Danh sách các sách nhận cùng thông tin liên quan
- Thao tác người dùng
    + Nhập thông tin về việc cho nhận sách
    + Yêu cầu cho nhận sách

- Màn hình Trả sách:
- Nộidung:
    + Ngày trả sách
    + Thông tin về việc trả sách
- Thao tác người dùng
    + Nhập thông tin về việc trả sách
    + Yêu cầu trả sách

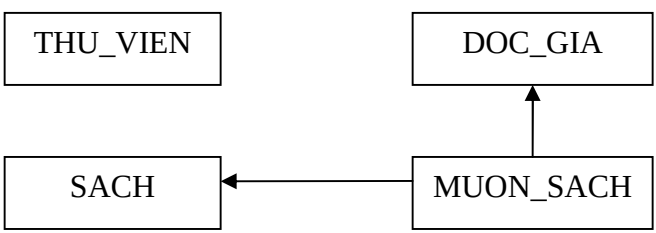
c. Hệ thống các hàm xử lý

- Hàm Lập thẻ: Kiểm tra tính hợp lệ và ghi nhận thẻ trên bộ nhớ phụ
- Hàm Tra cứu độc giả: Tìm thẻ độc giả theo các tiêu chuẩn khác nhau để cho phép cập nhật hay xóa thẻ
- Hàm Xóa thẻ: Xóa thẻ trên bộ nhớ phụ
- Hàm Nhập sách: Kiểm tra tính hợp lệ của sách và ghi nhận sách trên bộ nhớ phụ
- Hàm Xóa sách: Xóa sách trên bộ nhớ phụ
- Hàm Cho mượn sách: Kiểm tra tính hợp lệ của việc cho mượn sách và ghi nhận các thông tin cho mượn sách trên bộ nhớ phụ

- Hàm Tra cứu sách: Tìm sách theo các tiêu chuẩn khác nhau để cho phép cập nhật hay xóa sách
- Hàm Tính số sách độc giả đang mượng: Tính số lượng sách độc giả đang mượn và chưa trả
- Hàm Kiểm tra độc giả có sách mượn quá hạn: Kiểm tra độc giả có sách mượn quá hạn và trả về 1 nếu đúng, 0 nếu sai
- Hàm Kiểm tra tình trạng sách: Kiểm tra sách đang được mượn trả về 1 nếu đúng , 0 nếu sai
- Hàm Tra cứu phiếu cho mượn sách: Tra cứu các phiếu mượn sách theo nhiều tiêu chuẩn để cập nhật hay xó phiếu cho mượn
- Hàm Xóa phiếu cho mượn sách: Xóa thông tin về việc cho mượn sách trên bộ nhớ phụ
- Hàm Trả sách: Ghi nhận việc trả sách trên bộ nhớ phụ
- Hàm Tính tiền phạt: Tính tiền phạt khi độc giả trả sách trễ hạn

d. Hệ thống các bảng dữ liệu:

- **Sơ đồ logic**

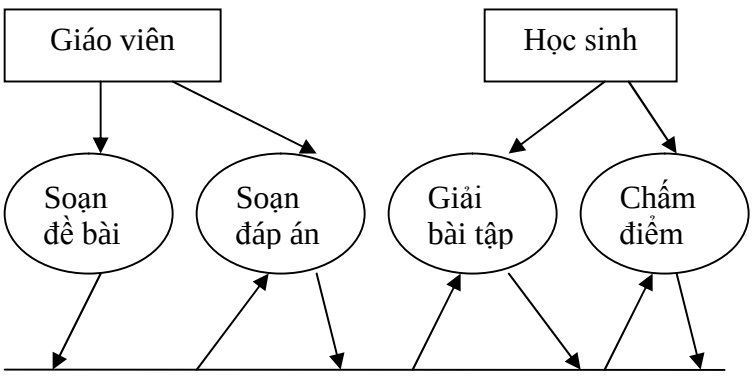


- Bảng THU_VIEN: các thông tin về thư viện
- Bảng DOC_GIA: Các thông tin về độc giả
- Bảng SACH: Các thông tin về sách
- Bảng MUON_SACH: Các thông tin về mượn trả sách

Ví dụ 2: Xét phần mềm hỗ trợ giải bài tập phương trình đại số với 4 yêu cầu

- 1. Soạn đề bài
- 2. Soạn đáp án
- 3. Giải bài tập
- 4. Chấm điểm

a. Mô hình hóa yêu cầu



b. Thiết kế phần mềm

- Màn hình chính:
- Nộidung:
  + Thông tin về sách bài tập
  + Thông tin về các bài tập của sách
- Thao tác người dùng: Tra cứu và chọn bài tập

- Màn hình Soạn đề bài:
- Nộidung:
  + Thông tin về đề bài
- Thao tác người dùng
  + Nhập thông tin về đề bài
  + Yêu cầu phát sinh đề
  + Yêu cầu ghi nhận đề

- Màn hình Soạn đáp án:
- Nộidung:
  + Thông tin về đáp án

- Thao tác người dùng
    + Nhập thông tin về đáp án
    + Yêu cầu ghi nhận đáp án

▪ Màn hình Nhận bài giải:
- Nộidung:
    + Thông tin về bài giải
- Thao tác người dùng
    + Nhập thông tin về bài giải
    + Yêu cầu ghi nhận bài giải
    + Yêu cầu chấm điểm

▪ Màn hình Chấm điểm:
- Nộidung:
    + Thông tin về bài giải
    + Thông tin về việc chấm điểm
    + Thông tin về đáp án
- Thao tác người dùng
    + Xem thông tin điểm
    + Yêu cầu xem đáp án

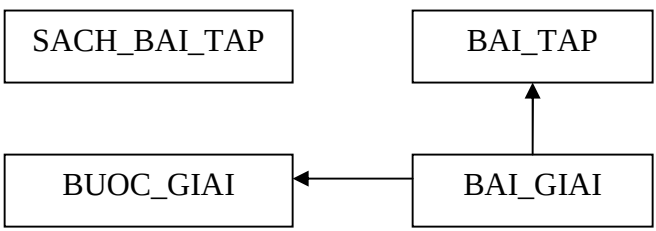
c. Hệ thống các hàm xử lý:

- Hàm Soạn thảo đề bài: Ghi nhận đề bài của giáo viên trên bộ nhớ phụ (giới hạn không kiểm tra tính hợp lệ của đề bài)
- Hàm Tra cứu bài tập: Tìm kiến bài tập theo nhiều tiêu chuẩn khác nhau để có thể cập nhật, xóa hay soạn đáp án
- Hàm Xóa bài tập : Xóa bài tập trên bộ nhớ phụ
- Hàm Soạn đáp án: Kiểm tra tính hợp lệ của đáp án của giáo viên và ghi nhận đáp án trên bộ nhớ phụ
- Hàm Xóa đáp án: Xóa bài tập trên bộ nhớ phụ
- Hàm Ghi nhận bài giải: Kiểm tra tính hợp lệ bài giải của học sinh và ghi nhận bài giải trên bộ nhớ phụ
- Hàm Biến đổi: Biến đổi một biểu thức thành một đa thức

- Hàm Khai triển: Nhân đa thức
- Hàm Rút gọn: Cộng 2 đa thức
- Hàm so sánh: So sành đa thức
- Hàm Xóa bài giải: Xóa bài giải của học sinh trên bộ nhớ phụ
- Hàm Chấm điểm: Tính điểm số bài giải của học sinh
- Hàm Xem đáp án: Trình bày các bước giải của đáp án cho học sinh xem

d. Hệ thống lưu trữ

- Sơ đồ logic



- Bảng SACH_BAI_TAP: các thông tin về sách bài tập
- Bảng BAI_TAP: Các thông tin về các bài tập của sách
- Bảng BUOC_GIAI: Các thông tin về các bước giải trong một bài giải
- Bảng BAI_GIAI: Các thông tin về đáp án và các bài giải của một bài tập

Chương 4: THIẾT KẾ DỮ LIỆU

1. Tổng quan

Mục tiêu chính của thiết kế dữ liệu là mô tả cách thức tổ chức lưu trữ các dữ liệu của phần mềm. Có hai dạng lưu trữ chính mà người thiết kế cần phải cân nhắc và lựa chọn.

- Lưu trữ dưới dạng tập tin
- Lưu trữ dưới dạng cơ sở dữ liệu

Lưu trữ dưới dạng tập tin thường chỉ thích hợp với một số phần mềm đặc thù (cờ tướng, trò chơi, v.v.) đặc điểm chung của các phần mềm này là chú trọng rất nhiều vào xử lý, hình thức giao diện và không chú trọng nhiều đến việc lưu trữ lại các thông tin được tiếp nhận trong quá trình sử dụng phần mềm (thông thường các thông tin này được tiếp nhận và xử lý ngay).

Cách tiếp cận dùng cơ sở dữ liệu rất thông dụng và giáo trình này sẽ giới hạn trình bày chi tiết các phương pháp kỹ thuật liên quan đến việc tổ chức lưu trữ dữ liệu dùng cơ sở dữ liệu quan hệ. Giáo trình này sẽ không nhắc lại các khái niệm cơ bản về cơ sở dữ liệu và giả sử rằng người xem đã biết qua các khái niệm này. Tuy nhiên chúng ta cũng nên xem lại các bước để hình thành nên mô hình dữ liệu quan hệ trong quá trình thiết kế dữ liệu

2. Kết quả của thiết kế

Cách thức tổ chức lưu trữ dữ liệu của phần mềm được mô tả thông qua 2 loại thông tin sau:

- Thông tin tổng quát

  Cung cấp góc nhìn tổng quan về các thành phần lưu trữ
  - Danh sách các bảng dữ liệu: Việc lưu trữ cần sử dụng bao nhiêu bảng dữ liệu và đó là các bảng nào ?

- Danh sách các liên kết: Các bảng dữ liệu có quan hệ, có mối liên kết giữa chúng ra sao?

- Thông tin chi tiết:
  - Danh sách các thuộc tính của từng thành phần: Các thông tin cần lưu trữ của thành phần ?
  - Danh sách các Miền giá trị toàn vẹn: Các qui định về tính hợp lệ của các thông tin được lưu trữ

Có nhiều phương pháp, nhiều đề nghị khác nhau về việc mô tả các thông tin trên. Giáo trình này chọn sơ đồ logic để biểu diễn các thông tin tổng quát và bảng thuộc tính. Miền giá trị để mô tả chi tiết các thành phần trong sơ đồ logic.

Sơ đồ logic là sơ đồ cho phép thể hiện hệ thống các bảng dữ liệu cùng với quan hệ mối nối liên kết giữa chúng. Các ký hiệu được dùng trong sơ đồ rất đơn giản như sau:

Bảng: hình chủ nhật

Liên kết: (xác định duy nhất một): Mũi tên

Mũi tên hình trên có ngữ nghĩa: 1 phần tử A sẽ xác định duy nhất 1 phần tử B, ngược lại 1 phần tử B có thể tương ứng với nhiều phần tử A.

Ví dụ: Với phần mềm quản lý thư viện có sơ đồ logic sau:

Theo sơ đồ này việc lưu trữ các dữ liệu của phần mềm quản lý thư viện được tổ chức 3 bảng (DOCGIA, MUONSACH, SACH) vùng với 2 liên kết giữa chúng

Tất nhiên sơ đồ trên chỉ là một trong các cách thức tổ chức lưu trữ dữ liệu còn nhiều cách khác có thể có. Chi tiết các cách này sẽ được trình bày trong phương pháp thiết kế cơ sở dữ liệu.

Bảng thuộc tính cho phép mô tả chi tiết thành phần trong sơ đồ logic theo dạng như sau:

- Thành phần
- Ý nghĩa

STT	Thuộc tính	Kiểu	Miền giá trị	Ý nghĩa	Ghi chú
1					
2					
…					

Bảng miền giá trị cho phép mô tả các Miền giá trị giữa các thuộc tính cùng một thành phần hay nhiều thành phần khác nhau.

MSố	Mô tả miền giá trị	Thành phần liên quan	Ghi chú
RB1			
RB2			
…			

Ví dụ:

Ghi chú:

- Bảng thuộc tính cho phép mô tả chi tiết thanh phần cần lưu trữ và sẽ được dùng trong báo cáo về thiết kế dữ liệu của phần mềm. Tuy nhiên cách mô tả trên khá dài dòng, trong giáo trình này sẽ sử dụng một dạng trình bày cô đọng hơn theo dạng lược đồ quan hệ. Với dạng trình bày này gồm tên bảng và thuộc tính đi kèm, các thuộc tính khóa được gạch chân.

Ví dụ:

DOC_GIA(MDG,Hoten,Loaidg,Ngsinh, Nglapthe, Diachi)

SACH(MSACH,Tensach,Theloai, NgNhap, Tacgia, Nhaxb, Namxb)

MUON(MDG,MSACH,NgMuon,Ngtra)

4. Quá trình thiết kế

Có 2 cách tiếp cận chính để thiết kế dữ liệu:

- **Phương pháp trực tiếp:**

  Từ các yêu cầu đã xác định, tạo lập trực tiếp sơ đồ logic cùng với bảng thuộc tính, bảng miền giá trị. Các tiếp cận này rất khó thực hiện đối với sơ đồ logic phức tạp.

- **Phương pháp gián tiếp:**

  Từ các yêu cầu đã xác định, tạo lập mô hình quan niệm dữ liệu, và sau đó đưa vào mô hình này sẽ tạo lập sơ đồ logic, bảng thuộc tính, bảng miền giá trị. Các tiếp cận này dễ thực hiện hơn vì mô hình quan niệm dữ liệu thường đơn giản (chứa các thành phần dữ liệu bản chất nhất của phần mềm). Khái niệm chi tiết về mô hình quan niệm dữ liệu cùng với các bước cụ thể sẽ được trình bày chi tiếp trong phần sau.

Tương ứng với 3 yêu cầu của phần mềm, quá trình thiết kế dữ liệu bao gồm 3 bước lớn:

- Thiết kế với tính đúng đắn
- Thiết kế với yêu cầu chất lượng
- Thiết kế với yêu cầu hệ thống

- *Thiết kế với tính đúng đắn*

- Đảm bảo đầy đủ và chính xác về mặt ngữ nghĩa các thông tin liên quan đến các công việc trong yêu cầu.
- Các thông tin phục vụ cho các yêu cầu chất lượng sẽ không được xét đến trong bước thiết kế này.

- *Thiết kế với yêu cầu chất lượng*

- Vẫn đảm bảo tínhh đúng đắn nhưng thỏa mãn thêm các yêu cầu chất lượng khác (tiến hóa, tốc độ nhanh, lưu trữ tối ưu).
- Cần chú ý bảo đảm tính đúng đắn khi cải tiến sơ đồ logic.

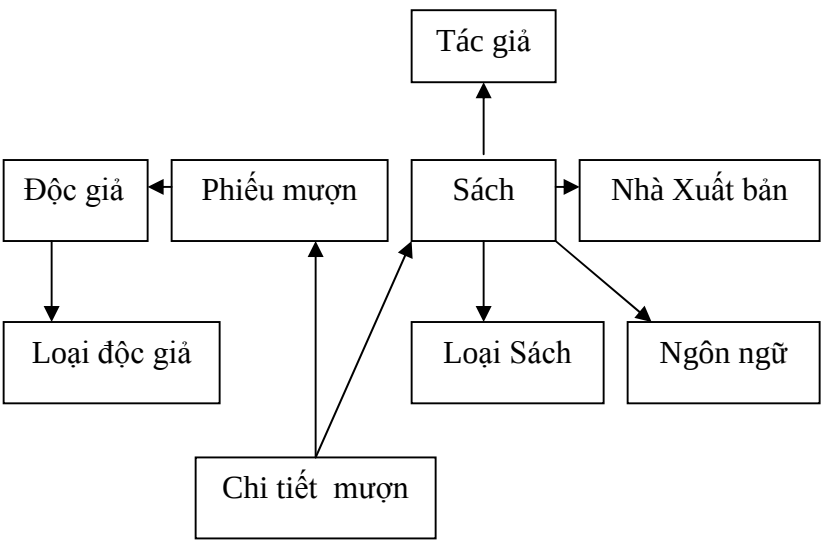
▪ *Thiết kế với yêu cầu hệ thống*

- Vẫn đảm bảo tính đúng đắn và các yêu cầu chất lượng khác nhưng thỏa mãn thêm các yêu cầu hệ thống (phân quyền, cấu hình phần cứng, môi trường phần mềm, v.v)

Ví dụ: phần mềm quản lý thư viện:

Với phương pháp trực tiếp sẽ cho kết qủa như sau:

Sơ đồ logic:



Các bảng thuộc tính:

DOC_GIA**(**MDG,MLDG,HoTen,NgaySinh,DiaChi,DienThoai)

SACH(MSACH,MTG,MNXB,MLSACH,MNN,TenSach, Ngàymua, SoTrang)

PHIEU_MUON(MPHM, NgayMuon)

CHITIETMUON(MPHM, MSACH, NgayTra)

LOAISACH(MLSACH,TenLS,GhiChu)

LOAIDOCGIA(MLDG,TenLDG,GhiChu)

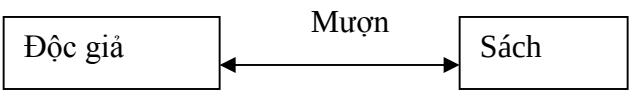
NHAXB(MNXB,TenNXB, GhiChu)

TACGIA(MTG,Ten, Ghichu)

NGONNGU(MNN,Ten,Ghichu)

Với phương pháp gián tiếp, ngoài kết qủa cuối cùng tương tự như phương pháp trực tiếp, còn có kết quả trung gian là mô hình quan niệm dữ liệu như sau:

+ Sơ đồ lớp đối tượng với 2 đối tượng chính Sách, Độc giả và 1 quan hệ Mượn giữa 2 lớp đối tượng trên

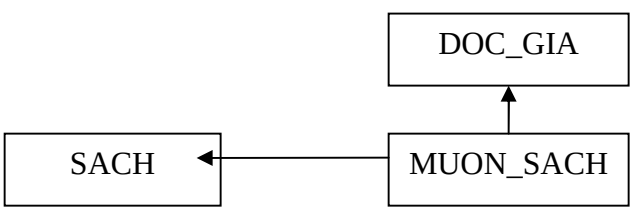


+ Mô hình chi tiết các thành phần trong sơ đồ lớp: **Xem chi tiết ở phụ lục B**

Ví dụ : Xét phần mềm với 4 yêu cầu: Lập thẻ độc giả, Nhận sách, Cho mượn sách, Trả sách

Thiết kế dữ liệu với tính đúng đắn

Sơ đồ logic



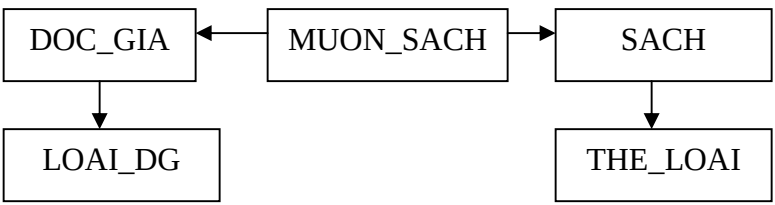
Chi tiết các bảng

DOCGIA**(**MDG,MLDG,HoTen,NgaySinh,DiaChi,DienThoai)

SACH(MSACH,MTG,MNXB,MLSACH,MNN,TenSach, Ngàymua, SoTrang)

MUON_SACH(MDG,MSACH,NgMuon,NgTra,Tienphat)

Thiết kế dữ liệu với tính tiến hóa
Sơ đồ logic



Chi tiết các bảng:

DOC_GIA**(**MDG,MLDG,HoTen,NgaySinh,DiaChi,DienThoai Ng_lapthe,Ng_hethan)

SACH(MSACH,Tensach,MTL,ng_Nhap, Tacgia,NamXB, NhaXB)

MUON_SACH(MDG,MSACH,NgMuon,NgTra,Tienphat

THE_LOAI(MTL,Tentheloai,GhiChu)

LOAI_DG(MLDG,TenLDG,GhiChu)

Thiết kế với tính hiệu quả (truy xuất nhanh)
Sơ đồ logic

Cũng với sơ đồ logic như trên nhưng ta có các bảng thuộc tính:

DOC_GIA**(**MDG,MLDG,HoTen,NgaySinh,DiaChi,DienThoai Ng_lapthe,Ng_hethan, SosachMuon, TinhTrangtra)

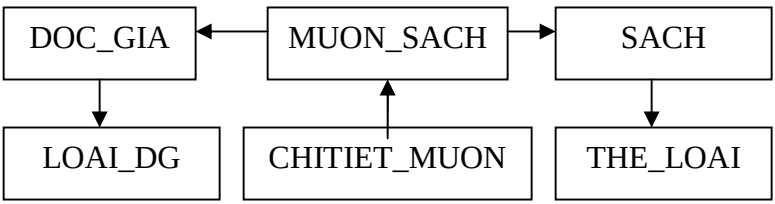
SACH(MSACH,Tensach,MTL,ng_Nhap, Tacgia,NamXB, NhaXB, TinhTrangMuon)

MUON_SACH(MDG,MSACH,NgMuon,NgTra,Tienphat)

THE_LOAI(MTL,Tentheloai,GhiChu)

LOAI_DG(MLDG,TenLDG,GhiChu)

Thiết kế dữ liệu với tính hiệu quả (lưu trữ tối ưu)
Sơ đồ logic



Chi tiết các bảng thuộc tính

DOC_GIA**(**MDG,MLDG,HoTen,NgaySinh,DiaChi,DienThoai Ng_lapthe,Ng_hethan, SosachMuon, TinhTrangtra)

SACH(MSACH,Tensach,MTL,ng_Nhap, Tacgia,NamXB, NhaXB, TinhTrangMuon)

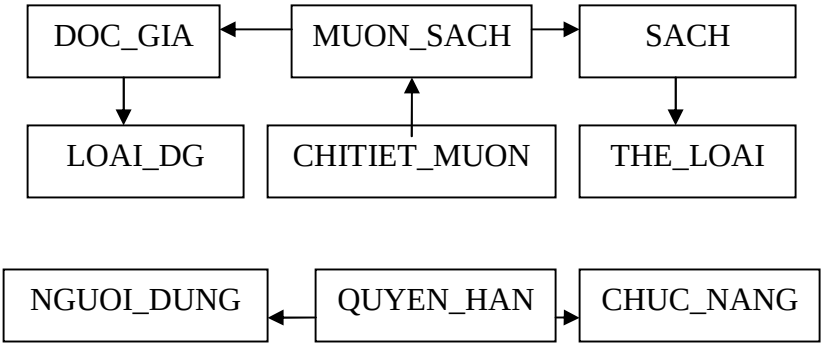
MUON_SACH(MDG,MSACH,NgMuon,NgTra,Tienphat)

CHITIET_MUON(MMUON,MSACH,NgTra,Tienphat)

THE_LOAI(MTL,Tentheloai,GhiChu)

LOAI_DG(MLDG,TenLDG,GhiChu)

Thiết kế dữ liệu với yêu cầu phân quyền hệ thống (phân quyền)
Sơ đồ logic



Chi tiết các bảng

DOC_GIA(MDG,MLDG,HoTen,NgaySinh,DiaChi,DienThoai Ng_lapthe,Ng_hethan, SosachMuon, TinhTrangtra)

SACH(MSACH,Tensach,MTL,ng_Nhap, Tacgia,NamXB, NhaXB, TinhTrangMuon)

MUON_SACH(MDG,MSACH,NgMuon,NgTra,Tienphat)

CHITIET_MUON(MMUON,MSACH,NgTra,Tienphat)

THE_LOAI(MTL,Tentheloai,GhiChu)

LOAI_DG(MLDG,TenLDG,GhiChu)

NGUOI_DUNG(MND,HoTen, Ghichu)

CHUC_NANG(MCN,Ten_Chucnang, Ghichu)

QUYEN_HAN(MND,MCN)

4. Phương pháp thiết kế dữ liệu

4.1 Phương pháp trực tiếp

Bước 1:

- Lập sơ đồ với 1 thành phần duy nhất
- Đánh giá tính đúng đắn so với các yêu cầu và chuyển sang bước 2 nếu cần thiết

Bước 2:

- Tách 1 số thuộc tính để tạo ra các thành phần mới
- Xác định liên kết giữa các thành phần
- Đánh giá tính đúng đắn so với các yêu cầu và lặp lại bước 2 nếu cần thiết

Ví dụ: phần mềm quản lý thư viện

Cách 1: Chỉ dùng 1 thành phần SÁCH
Masach, Ten, Theloai, Ngaymua, Tacgia, NhaXB, NamXB
HotenDG, LoaiDG, Ngaylamthe, Ngaymuon, Ngaytra

Cách 2: Dùng 2 thành phần SACH,DOCGIA
Cách 2.1 : Chỉ lưu trữ lần mượn sách cuối cùng

SACH
MSACH, MADG, Ten, Theloai, NgayMua, TacGia, NhaXB, NamXB, Ngaymuon, NgayTra.
DOCGIA
MDG, HoTen, LoaiDG,Ngaylamthe,

Cách 2.2: Chỉ cho phép độc giả mượn tối đa 1 quyển sách

SACH
MSACH, Ten, Theloai, NgayMua, TacGia, NhaXB, NamXB, Ngaymuon, NgayTra.
DOCGIA
MDG, MSACH,HoTen, LoaiDG,Ngaylamthe, Ngaymuon
Cách 3: Dùng 3 thành phần SACH,DOCGIA, MUONSACH
SACH
MSACH, Ten, Theloai, NgayMua, TacGia, NhaXB, NamXB, Ngaymuon, NgayTra.
DOCGIA
MDG, HoTen, LoaiDG,Ngaylamthe,
MUONSACH
Mmuon,MDG,MSACH, Ngaymuon, Ngaytra

Ví dụ: Phần mềm quản lý học sinh

Cách 1: Dùng 1 thành phần HOCSINH

HOCSINH

MAHS, HoTen, Ngaysinh, GioiTinh, Lop, Monhoc, LoaiKT, HocKy,Diem, Ngayvang, Lydo

Cách 2: Dùng 3 bảng HOCSINH, KIEMTRA, DIEMDANH

HOCSINH

MAHS, Hoten, Ngaysinh, GioiTinh, Lop

KIEMTRA

MAKT,MAHS, Monhoc,LoaiKT,Hocky, Diem

DIEMDANH

MADD,MAHS,Ngayvang, Lydo

4.2 Phương pháp gián tiếp

Bước 1:

- Lập sơ đồ lớp
- Xác định các lớp đối tượng
- Xác định quan hệ giữa các lớp đối tượng và lập sơ đồ

Bước 2:

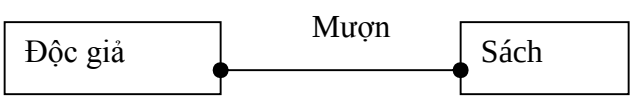
- Ánh xạ từ sơ đồ lớp vào sơ đồ logic
- Ánh xạ các lớp đối tượng
- Ánh xạ các quan hệ giữa các lớp đối tượng

Bước 3:

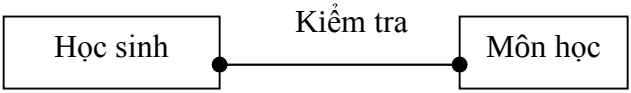
- Hoàn chỉnh sơ đồ logic
- Bổ sung các thành phần theo yêu cầu
- Mô tả chi tiết các thuộc tính của các thành phần

4.2.1 Lập sơ đồ lớp

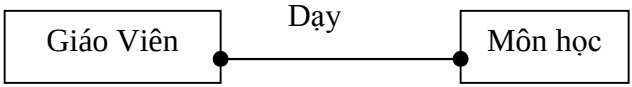
Ví dụ: Với phần mềm quản lý thư viện 2 đối tượng chính là Độc giả, Sách và quan hệ giữa chúng là quan hệ mượn sách



Với phần mềm quản lý học sinh trường phổ thông trung học 2 đối tượng chính là Học sinh, Môn học và quan hệ giữa chúng là quan hệ kiểm tra



Với phần mềm xếp thời khóa biểu trường trung học phổ thông 2 đối tượng chính là Giáo Viên, Môn học và quan hệ giữa chúng là quan hệ dạy.



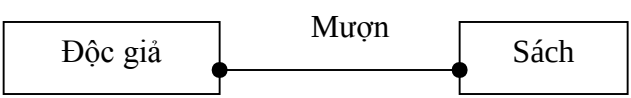
4.2.2 Ánh xạ sơ đồ lớp

Ánh xạ lớp đối tượng. Mỗi đối tượng trong sơ đồ lớp tương ứng với 1 thành phần trong sơ đồ logic

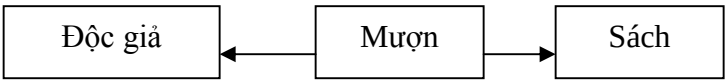
4.2.3 Ánh xạ quan hệ

- Quan hệ 1-n: Quan hệ 1-n trong sơ đồ lớp giữa 2 lớp đối tượng A,B (1 A nhiều B) tương ứng với liên kết xácđịnh duy nhất vừ A sang B trong sơ đồ logic.
- Quan hệ m-n: Quan hệ m-n C trong sơ đồ lớp giữa 2 lớp đối tượng A,B tương ứng với 1 thành phần C trong sơ đồ logic. Thành phần này có liên hệ xác định duy nhất A,B.

Sơ đồ lớp:



Sơ đồ logic:

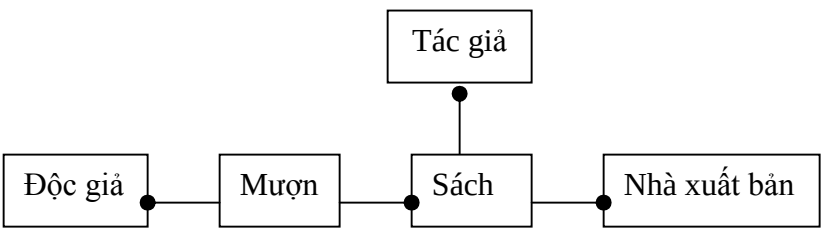


4.2.4 Hoàn chỉnh sơ đồ logic

1. Bổ sung các thành phần

+ Đối tượng phụ: Mỗi đối tượng phụ tương ứng với 1 thành phần trong sơ đồ logic

+ Các thành phần khác: Xem xét lại tính đúng đắn và bổ sung thêm nếu cần thiết



2. Mô tả chi tiết thuộc tính các thành phần

+ Thuộc tính khóa chính:

- Mỗi thành phần ứng với đối tượng (chính, phụ) cần 1 thuộc tính khóa riêng)

- Các thành phần còn lại, tùy theo ý nghĩa sử dụng sẽ có thuộc tính khóa riêng hay dùng tổ hợp thuộc tính khóa của các thành phần khác

Ví dụ: Các thành phần Độcgiả, Sách, Nhà xuất bản, Tác giả sẽ có thuộc tính khóa chính tương ứng là MDG, MSACH, MNXB, MTG.

Thành phần mượn cũng sẽ có khóa chính là MMUON (không dùng tổ hợp các thuộc tính khóa ngoại được ?)

+ Thuộc tính khóa ngoại:

- Thể hiện đúng liên kết giữa các thành phần trong sơ đồ logic: nếu A xác định duy nhất B thì A có thuộc tính là khoá chính của B (đó là khóa ngoại của A)

Ví dụ:

Thành phần Mượn có 2 khóa ngoại: MDG,MSACH
Thành phần Sách có 2 khoá ngoại: MNXB, MTG, MDG

+ Các thuộc tính khác:

Dựa vào yêu cầu lưu trữ, chú ý các loại thuộc tính sau:

- Định danh: Tên
- Loại: Sự phân loại
- Thời gian: Ngày tháng
- Không gian: vị trí
- Định lượng: độ đo, tính chất, v.v.v

Ví dụ: Độc giả sẽ có thuộc tính khác như:

HoTen (định danh)
LoaiDG (loại)
Ngaysinh (thời gian)
Ngayhethan (thời gian)
Diachi (không gian)

Sách sẽ có thuộc tính khác như:

TenSach (định danh)
LoaiSach (loại)
NgayMua (thời gian)
GiaTien (định lượng)

5. Thiết kế dữ liệu với tính đúng đắn

- Các bước thực hiện:

Bước 1: Chọn một yêu cầu và xác định sơ đồ logic cho yêu cầu đó

Bước 2: Bổ sung thêm một yêu cầu và xem xét lại sơ đồ logic

+ Nếu sơ đồ logic vẫn đáp ứng được thì tiếp tục bước 3

+ Nếu sơ đồ logic không đáp ứng được thì bổ sung vào sơ đồ thuộc tính mới (ưu tiên 1) hoặc thành phần mới (ưu tiên 2) cùng với các thuộc tính và liên kết tương ứng

Bước 3: Quay lại bước 2 cho đến khi đã xem xét đầy đủ các yêu cầu

Ghi chú:

- Với mỗi yêu cầu cần xác định rõ cần lưu trữ các thông tin gì? dựa vào luồng dữ liệu đọc/ghi trong sơ đồ luồng dữ liệu tương ứng) và tìm cách bổ sung các thuộc tính để lưu trữ các thông tin này
- Chỉ xem xét tính đúng đắn
- Cần chọn các yêu cầu theo thứ tự từ đơn giản đến phức tạp (thông thường yêu cầu tra cứu là đơn giản nhất)
- Với yêu cầu phức tạp có thể phải bổ sung vào sơ đồ logic nhiều thành phần mới

Khóa của các thành phần phải dựa trên ngữ nghĩa tương ứng trong thế giới thực

6. Thiết kế dữ liệu và yêu cầu chất lượng

- **Mục tiêu**

Xem xét đánh giá sơ đồ logic theo các yêu cầu về chất lượng và tiến hành cập nhật lại sơ đồ để bảo đảm các tiêu chuẩn về chất lượng. Ngoài tính đúng đắn cần ưu tiên hàng đầu xem xét sự hơn kém nhau giữa các phần mềm chính là mức độ thỏa mãn các tiêu chuẩn chất lượng còn lại (đặc biệt là tính tiến hóa).

6.1 Xem xét tính tiến hóa

Để bảo đảm tính tiến hóa, sơ đồ logic sẽ còn bổ sung cập nhật lại nhiều thành phần qua các bước thiết kế chi tiết. Trong các bước đầu tiên là thiết kế dữ liệu, chúng sẽ giới hạn xem xét đến các thuộc tính có giá trị rời rạc.

Thuộc tính có giá trị rời rạc là các thuộc tính mà miền giá trị chỉ bao gồm một số giá trị nhất định. Các giá trị này thông thường thuộc về tập hợp có độ biến động rất ít trong quá trình sử dụng phần mềm.

Ví dụ:

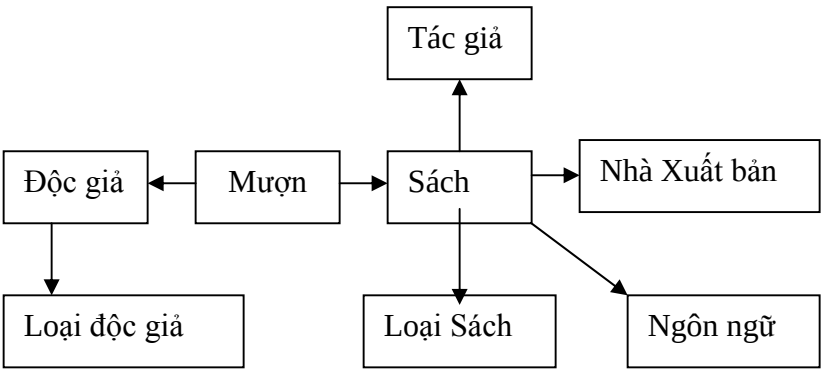
LOAIDG (thành phần độc giả): Thư viện hiện tại chỉ có 3 loại độc giả là ‘A’, ‘B’,’C’ và khả năng có thêm loại độc giả mới rất thấp.

Ngôn ngữ (thành phần Sách): Các sách trong thư viện hiện tại chỉ có 3 loại ngôn ngữ ‘Việt’, ‘Anh’, ‘Pháp’ và khả năng thêm sách thuộc ngôn ngữ mới rất thấp.

Tuy nhiên cần lưu ý rằng khả năng biến động trên tập hợp giá trị của thuộc tính rời rạc là thấp nhưng không phải là không có. Và khi xảy ra biến động (thêm loại độc giả, thêm sách thuộc ngôn ngữ mới) nếu không chuẩn bị trước trong thiết kế thì người dùng sẽ không thể khai báo được các biến động này với phần mềm, và do đó có thể một số chức năng sẽ không thực hiện được (ví dụ khong thể thêm sách mới với ngôn ngữ tiếng Hoa).

Để chuẩn bị tốt cho biến động về sau (nếu có) trong tập hợp các giá trị của thuộc tính rời rạc. Chúng ta sẽ tách các thuộc tính này thành một thành phần trong sơ đồ logic. Khi đó người dùng trong quá trình sử dụng hoàn toàn có thể cập nhật lại tập hợp các giá trị này tương ứng với các biến động thực tế trong thế giới thực.

Sơ đồ logic khi tách các thuộc tính rời rạc như sau:



6.2 Xem xét tính hiệu quả (tốc độ)

- Phạm vi xem xét:
  - Chỉ giới hạn xem xét việc tăng tốc độ thực hiện của phần mềm bằng cách bổ sung thêm các thuộc tính vào các bảng dùng lưu trữ các thông tin đã tính toán trước (theo qui tắc nào đó từ các thông tin gốc đã được lưu trữ)

    Ví dụ: số sách đang mượn của độc giả
  - Các thông tin này phải được tự động cập nhật khi có bất kỳ thay đổi thông tin gốc liên quan

    Ví dụ độc giả mượn thêm hoặc trả sách

    Học sinh có thêm cột điểm
- Các bước tiến hành:

- Bước 1: Chọn một yêu cầu và xem xét cần bổ sung thông tin gì trên bộ nhớ phụ để tăng tốc độ thực hiện của xử lý liên quan (các thông tin xử lý phải đọc mà không cần thực hiện việc tính toán)

- Bước 2: Quay lại bước 1 cho đến khi đã xem xét đầy đủ các yêu cầu

Ghi chú:

- Sau mỗi bước nhất thiết phải lập bảng danh sách các thuộc tính tính toán cùng với thông tin liên quan

  + Thông tin gốc

  + Xử lý tự động cập nhật thông tin gốc (chi tiết về các xử lý này sẽ được mô tả trong phần thiết kế xử lý)

- Nếu thông tin gốc thường xuyên bị thay đổi, việc bổ sung thuộc tính tính toán để tăng tốc độ thực hiện sẽ mất ý nghĩa (thậm chí theo chiều ngược lại)

- Việc tăng tốc độ truy xuất có thể sẽ dẫn đến việc lưu trữ không tối ưu

- Thứ tự xem xét các yêu cầu theo thứ tự từ đầu đến cuối (không cần chọn như các bước trong thiết kế dữ liệu)

  Ví dụ: Phần mềm quản lý giải vô địhc bóng đá quốc gia với bảng thuộc tính tính toán

  Thuộc tính: Tong_ban_thang, Tong_the_phat, Diem_so là những thuộc tính có thể xử lý tự động cập nhật

6.3 Xem xét tính hiệu quả (lưu trữ)

Tính hiệu quả trong thiết kế dữ liệu sẽ được xem xét dưới góc độ lưu trữ tối ưu. Vấn đề đặt ra là xây dựng sơ đồ logic sao cho vẫn bảo đảm lưu trữ đầy đủ thông tin theo yêu cầu nhưng với dung lượng lưu trữ nhỏ nhất có thể có. Vấn đề này đặc biệt quan trọng với các phần mềm với hệ thống lưu trữ lớn và nhiều phát sinh thông tin cần lưu trữ theo thời gian.

Khi đó cần đặc biệt quan tâm đến các thành phần mà dữ liệu tương ứng được phát sinh nhiều theo thời gian. Chúng ta sẽ tìm cách bố trí lại sơ đồ logic sao cho vẫn đảm bảo thông tin mà dung lượng lưu trữ lại ít hơn.

- Các bước tiến hành:

Bước 1:Lập danh sách các bảng cần được xem xét để tối ưu hóa việc lưu trữ

- Xem xét và xác định các công việc có tần suất thực hiện thường xuyên và bổ sung vào danh sách chọn các bảng được sử dụng tương ứng của công việc này
- Xem xét các bảng mà khóa của bảng bao gồm nhiều thuộc tính và bổ sung bảng này vào danh sách được chọn

Bước 2: Tối ưu hóa việc lưu trữ các bảng có khối lượng dữ liệu lưu trữ lớn thông qua việc tối ưu hóa lưu trữ từng thuộc tính trong bảng

- Xác định các thuộc tính mà việc lưu trữ chưa tối ưu. Ưu tiên xem xét các thuộc tính có kiểu chuỗi
- Tối ưu hóa việc lưu trữ tùy theo từng trường hợp cụ thể
- Một trong các trường hợp thông dụng nhất là chuỗi có kích thước lớn và giá trị được sử dụng nhiều lần trong các mẫu tin khác nhau (ví dụ: thuộc tính tác giả, Nha_xb trong bảng SACH của phần mềm quản lý sách)
- Với trường hợp trên việc tối ưu hoá có thể thực hiện thông qua việc bổ sung các bảng mới (bảng TAC_GIA, NHA_XB) và tổ chức cấu trúc bảng SACH (thay thuộc tính TAC_GIA bằng MTG, thay thuộc tính NHA_XB bằng MNXB)

Bước 3: Tối ưu hóa các bảng mà khóa của bảng bao gồm nhiều thuộc tính.

Phân rã bảng đang xét thành hai bảng. Trong đó, một bảng chứa các thuộc tính mà giá trị được lặp lại nhiều lần trong cùng một lần thực hiện công việc tương ứng trong thế giới thực. Bảng này cần có khóa riêng (sẽ được bảng còn lại sử dụng để tham chiếu đến)

Ghi chú:

- Việc phân rã giúp cho lưu trữ tối ưu tuy nhiên:
    - Tốc độ truy xuất có thể sẽ chậm hơn
    - Việc thực hiện xử lý khó khăn hơn (thuật giải phức tạp hơn)
- Cần cân nhắc trước khi thực hiện phân rã
- Việc đánh giá khóa riêng cho bảng đã phân ra có thể cần kiểm tra thêm số phụ thuộc hàm

Ví dụ minh họa: Phần mềm quản lý bán sách

Bước 1: Các bảng cần xem xét

NHAP_SACH(MSACH, Ng_Nhap, So_luong, Don_gia, Thanh_tien)

HOA_DON(MHD,MSACH, Khach_hang, Ng_lap_hd, So_Luong, Don_gia, Thanh_tien)

Bước 2:

- Bổ sung bảng KHACH_HANG

KHACH_HANG(MKH,Ho_ten,Ghi_chu)

- Tổ chức lại bảng HOA_DON

HOA_DON (MHD,MSACH,MKH,Ng_lap_hd,So_luong, Don_gia, Thanh_tien)

Bước 3:

- Phân rã bảng NHAP_SACH thành 2 bảng NHAP_SACH, CT_NHAP

NHAP_SACH(MNHAP,Ng_Nhap)

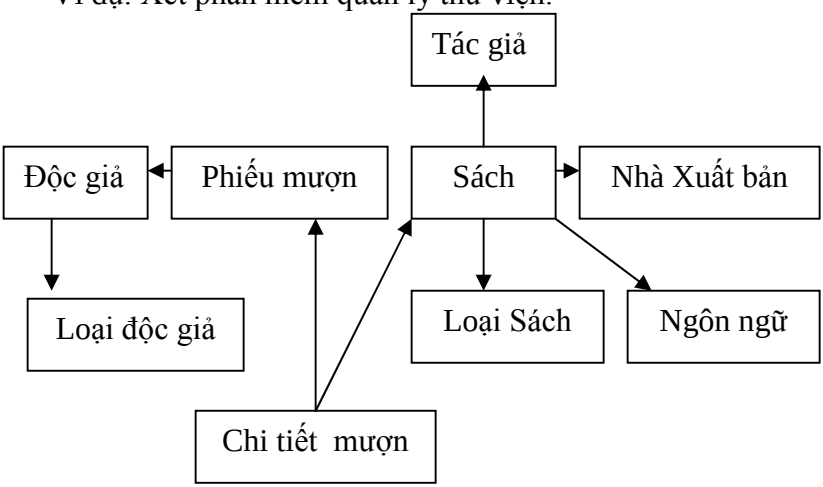
CT_NHAP(MNHAP,MSACH,So_luong, Don_gia, Thanh_tien)

- Phân rã bảng HOA_DON thành 2 bảng HOA_DON, CT_HOA_DON

HOA_DON(MHD,MKH, Ng_lap_hd)

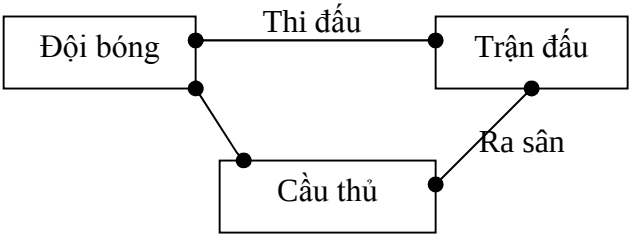
CT_HD(MHD, MSACH,So_luong, Don_gia, Thanh_tien)

Ví dụ: Xét phần mềm quản lý thư viện.



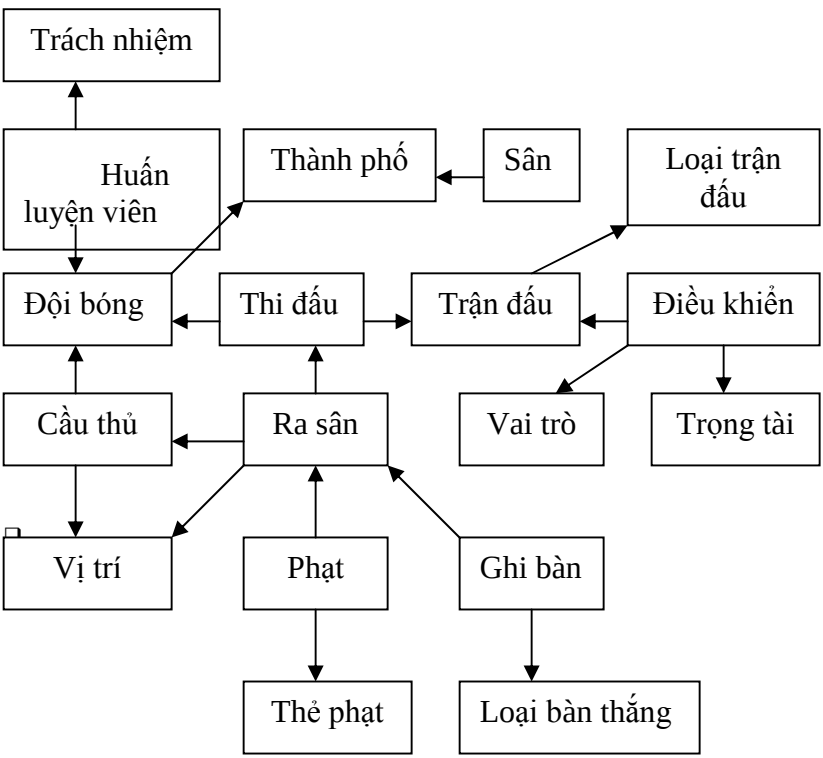
Ví dụ: Xét phần mềm quản lý giải bóng đá

Sơ đồ lớp



Mô tả chi tiết các thuộc tính: **Xem chi tiết phụ lục B**

- **Sơ đồ logic**



- **Mô tả chi tiết thuộc tính:** Xem chi tiết phụ lục B