

CÔNG NGHỆ PHẦN MỀM

SOFTWARE ENGINEERING

TS. Võ Đức Hoàng

Khoa Công nghệ thông tin

Trường Đại học Bách khoa - Đại học Đà Nẵng

- Chương 1: Giới thiệu Công nghệ phần mềm
- Chương 2: Các mô hình phát triển phần mềm
- Chương 3: Phân tích và đặc tả yêu cầu
- Chương 4: Các kỹ thuật đặc tả
- Chương 5: Thiết kế
- Chương 6: Lập trình và ngôn ngữ lập trình
- Chương 7: Kiểm thử
- Chương 8: Quản trị dự án

■ ***Giáo trình chính:***

- Lê Đức Trung, Công nghệ phần mềm, NXB KHKT, 2001.
- Ian Sommerville, Software Engineering, 10th Edition, Pearson, 2016

■ ***Tài liệu tham khảo:***

- Ronald Leach, Introduction to Software Engineering, CRC Press, 1999.
- Roger S. Pressman, Software Engineering : a practitioner's approach, 7th Edition, The McGraw-Hill Companies, Inc

Các kỹ thuật đặc tả (4)

- Khái niệm đặc tả
- Tại sao phải đặc tả?
- Phân loại các kỹ thuật đặc tả
- Các kỹ thuật đặc tả

Khái niệm đặc tả (specification)



- Đặc tả là mô tả ngắn gọn, chính xác vấn đề nào đó.
- Định nghĩa một hệ thống, mô-đun hay một sản phẩm cần phải **làm cái gì?**
- Không mô tả nó phải làm như thế nào
- Mô tả những **tính chất của vấn đề** đặt ra
- Không mô tả những tính chất của giải pháp cho vấn đề đó

Khái niệm đặc tả (specification)



- Đặc tả phần mềm là một tài liệu quan trọng trong quá trình phát triển phần mềm, nó định nghĩa rõ ràng các yêu cầu và mục tiêu của phần mềm. Đây là cơ sở để các kỹ sư phần mềm thiết kế và xây dựng sản phẩm.
- Tài liệu này sẽ cung cấp thông tin cho stakeholder như khách hàng, nhà phát triển, quản lý dự án.
- Mục tiêu của tài liệu này là đảm bảo hệ thống sẽ đáp ứng đầy đủ và chính xác yêu cầu của khách hàng

Tầm quan trọng của đặc tả phần mềm



Tương ứng yêu cầu

Đặc tả phần mềm xác định chính xác những gì hệ thống cần làm, đảm bảo phù hợp với nhu cầu thực tế của người dùng.

Nền tảng cho thiết kế

Đặc tả cung cấp cơ sở để thiết kế, lập kế hoạch và xây dựng phần mềm một cách có hệ thống.

Quản lý rủi ro

Xác định và giải quyết các vấn đề tiềm ẩn ngay từ đầu giúp tránh được những rắc rối lớn trong quá trình phát triển.

Truyền đạt thông tin

Đặc tả giúp các bên liên quan như nhà phát triển, khách hàng, quản lý dự án hiểu rõ về sản phẩm cần xây dựng.

- Giới thiệu: Phần này để mô tả mục đích, phạm vi, bối cảnh của hệ thống phần mềm.
- Yêu cầu chức năng: Ở phần này, BA sẽ mô tả chi tiết các chức năng mà hệ thống cần thực hiện để đáp ứng yêu cầu của khách hàng.
- Yêu cầu phi chức năng: Đây là những yêu cầu không ảnh hưởng trực tiếp đến chức năng như bảo mật, hiệu suất, v.v.

- Yêu cầu giao diện người dùng: Phần này sẽ mô tả cách người dùng tương tác với hệ thống.
- Yêu cầu dữ liệu: Ở phần này, BA sẽ cung cấp cấu trúc và quy tắc xử lý dữ liệu để xác xác định dữ liệu mà hệ thống sẽ lưu trữ và xử lý.
- Yêu cầu hệ thống: Cuối cùng là BA phải đặc tả các yêu cầu liên quan đến cấu trúc phần cứng và phần mềm mà hệ thống sẽ tương tác.

Tại sao phải đặc tả

- Hợp đồng
 - Sự thống nhất giá người dùng và người phát triển sản phẩm
- Hợp thức hóa
 - Sản phẩm lúc ra phải thực hiện chính xác những gì mong muốn
- Trao đổi
 - Giữa người sử dụng và người phát triển giữa những người phát triển
- Tái sử dụng

Phân loại các kỹ thuật đặc tả



- Đặc tả phi hình thức (informal)
 - Ngôn ngữ tự nhiên tự do
 - Ngôn ngữ tự nhiên có cấu trúc
 - Các ký hiệu đồ họa
- Đặc tả nửa hình thức (semi-informal)
 - Trộn lẫn cả ngôn ngữ tự nhiên, các kí hiệu toán học và các kí hiệu đồ họa
- Đặc tả hình thức (formal)
 - Đặc tả bằng mô hình
 - Đặc tả bằng ngôn ngữ hình thức hoá
 - Đặc tả bằng công thức toán học

Đặc tả hình thức hay không hình thức?



■ Đặc tả hình thức

- Chính xác (toán học)
- Hợp thức hóa hình thức (công cụ hóa)
- Công cụ trao đổi: khó đọc, khó hiểu
- Khó sử dụng

■ Đặc tả phi hình thức

- Dễ hiểu, dễ sử dụng
- Mềm dẻo
- Thiếu sự chính xác
- Không chắc chắn, nhập nhằng

- Ứng dụng trong các giai đoạn sớm của tiến trình phát triển
- Hạn chế lỗi trong phát triển phần mềm
- Ứng dụng chủ yếu trong phát triển các hệ thống “quan trọng” (critical systems)
 - Hệ thống điều khiển
 - Hệ thống nhúng
 - Hệ thống thời gian thực

- Ví dụ: chức năng kiểm tra lỗi chính tả
 - Yêu cầu: thông báo các lỗi chính tả khi duyệt
 - Đặc tả:
 - Các lỗi chính tả được gạch đỏ bên dưới
 - Các lỗi soạn thảo được gạch xanh bên dưới
 - Lỗi chính tả:
 - * Từ đơn không có trong từ điển
 - Lỗi soạn thảo
 - * Thừa dấu cách
 - * Không viết hoa đầu câu

- Tài liệu đặc tả yêu cầu phần mềm thường gồm những mục chính
 - Giới thiệu: Phần này để mô tả mục đích, phạm vi, bối cảnh của hệ thống phần mềm.
 - Yêu cầu chức năng: Ở phần này, BA sẽ mô tả chi tiết các chức năng mà hệ thống cần thực hiện để đáp ứng yêu cầu của khách hàng.
 - Yêu cầu phi chức năng: Đây là những yêu cầu không ảnh hưởng trực tiếp đến chức năng như bảo mật, hiệu suất, v.v.

- Tài liệu đặc tả yêu cầu phần mềm thường gồm những mục chính
 - Yêu cầu giao diện người dùng: Phần này sẽ mô tả cách người dùng tương tác với hệ thống.
 - Yêu cầu dữ liệu: Ở phần này, BA sẽ cung cấp cấu trúc và quy tắc xử lý dữ liệu để xác định dữ liệu mà hệ thống sẽ lưu trữ và xử lý.
 - Yêu cầu hệ thống: Cuối cùng là BA phải đặc tả các yêu cầu liên quan đến cấu trúc phần cứng và phần mềm mà hệ thống sẽ tương tác.

- Một công ty đang phát triển một ứng dụng mới để quản lý khách hàng. Hãy xây dựng tài liệu đặc tả yêu cầu phần mềm?
- Tài liệu có thể có các mục sau:
 - Giới thiệu
 - Yêu cầu chức năng
 - Yêu cầu phi chức năng
 - Yêu cầu giao diện người dùng
 - Yêu cầu dữ liệu
 - Yêu cầu hệ thống

■ Giới thiệu

- Mục đích của ứng dụng: Ứng dụng sẽ được sử dụng để quản lý khách hàng, bao gồm việc tạo mới, cập nhật, và xóa khách hàng.
- Phạm vi của ứng dụng: Ứng dụng sẽ bao gồm các chức năng sau:
 - Tạo mới khách hàng
 - Cập nhật thông tin khách hàng
 - Xóa khách hàng
 - Tìm kiếm khách hàng
 - Xem danh sách khách hàng
- Bối cảnh của ứng dụng: Ứng dụng sẽ được sử dụng bởi nhân viên kinh doanh của công ty.

- Yêu cầu chức năng của tài liệu đặc tả yêu cầu phần mềm sẽ mô tả chi tiết các chức năng mà ứng dụng cần thực hiện. Trong ví dụ trên, các yêu cầu chức năng có thể bao gồm các yêu cầu sau:
 - Yêu cầu 1: Người dùng phải có thể tạo mới một khách hàng bằng cách cung cấp các thông tin sau:
 - Tên khách hàng
 - Địa chỉ khách hàng
 - Số điện thoại khách hàng
 - Email khách hàng
 - Yêu cầu 2: Người dùng phải có thể cập nhật thông tin của một khách hàng.
 - Yêu cầu 3: Người dùng phải có thể xóa một khách hàng.
 - Yêu cầu 4: Người dùng phải có thể tìm kiếm khách hàng bằng tên, địa chỉ, hoặc số điện thoại.
 - Yêu cầu 5: Người dùng phải có thể xem danh sách tất cả khách hàng.

- Yêu cầu phi chức năng của tài liệu đặc tả yêu cầu phần mềm sẽ mô tả các yêu cầu không ảnh hưởng trực tiếp đến chức năng của ứng dụng, chẳng hạn như bảo mật, hiệu suất, và khả năng sử dụng. Trong ví dụ trên, các yêu cầu phi chức năng có thể bao gồm các yêu cầu sau:
 - Yêu cầu 6: Ứng dụng phải được bảo mật bằng xác thực hai yếu tố.
 - Yêu cầu 7: Ứng dụng phải có khả năng xử lý 1000 giao dịch mỗi giây.
 - Yêu cầu 8: Ứng dụng phải dễ sử dụng và dễ học.

- Yêu cầu giao diện người dùng của tài liệu đặc tả yêu cầu phần mềm sẽ mô tả cách người dùng tương tác với ứng dụng. Trong ví dụ trên, các yêu cầu giao diện người dùng có thể bao gồm các yêu cầu sau:
 - Yêu cầu 9: Ứng dụng phải sử dụng giao diện đồ họa người dùng (GUI) trực quan và dễ sử dụng.
 - Yêu cầu 10: Ứng dụng phải cung cấp các lời nhắc và hướng dẫn rõ ràng cho người dùng.

- Yêu cầu dữ liệu của tài liệu đặc tả yêu cầu phần mềm sẽ cung cấp cấu trúc và quy tắc xử lý dữ liệu để xác định dữ liệu mà hệ thống sẽ lưu trữ và xử lý. Trong ví dụ trên, các yêu cầu dữ liệu có thể bao gồm các yêu cầu sau:
 - Yêu cầu 11: Ứng dụng sẽ lưu trữ dữ liệu khách hàng trong một cơ sở dữ liệu SQL.
 - Yêu cầu 12: Các trường dữ liệu của bảng khách hàng sẽ bao gồm:
 - Tên khách hàng (varchar(50))
 - Địa chỉ khách hàng (varchar(255))
 - Số điện thoại khách hàng (varchar(10))
 - Email khách hàng (varchar(255))

- Yêu cầu hệ thống của tài liệu đặc tả yêu cầu phần mềm sẽ đặc tả các yêu cầu liên quan đến cấu trúc phần cứng và phần mềm mà hệ thống sẽ tương tác. Trong ví dụ trên, các yêu cầu hệ thống có thể bao gồm các yêu cầu sau:
 - Yêu cầu 13: Ứng dụng sẽ được triển khai trên máy chủ Linux.
 - Yêu cầu 14: Ứng dụng sẽ sử dụng cơ sở dữ liệu SQL Server.

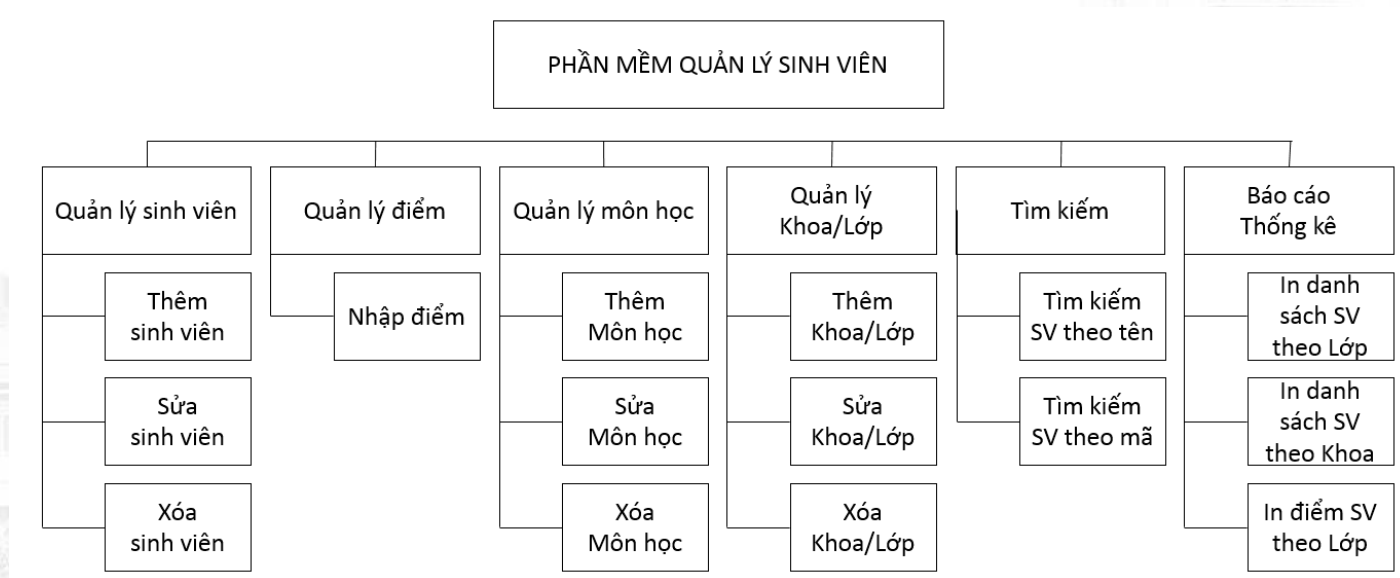
Các kỹ thuật đặc tả hình thức



- Biểu đồ phân rã chức năng
- Biểu đồ luồng dữ liệu
- Biểu đồ thực thể kết hợp
- Làm bản mẫu
- Máy trạng thái hữu hạn
- Mạng Petri
- Điều kiện trước sau
- Kiểu trừu tượng
- ~~■ Ngôn ngữ đặc tả Z~~

Biểu đồ phân rã chức năng

- Biểu đồ phân rã chức năng là một công cụ quản lý dự án quan trọng, giúp chia nhỏ các chức năng chính của một dự án thành các hoạt động và nhiệm vụ cụ thể. Nó cung cấp một cách nhìn tổng quan về phạm vi và quy mô của dự án, giúp lập kế hoạch và theo dõi tiến độ một cách hiệu quả.



Xác định các chức năng

Biểu đồ phân rã chức năng giúp xác định định và mô tả chi tiết các chức năng chính và phụ của một sản phẩm hoặc hệ thống.

Phân tích yêu cầu

Thông qua việc xây dựng biểu đồ, các yêu cầu và ràng buộc của sản phẩm có thể có thể được xác định rõ ràng.

Thiết kế hiệu quả

Biểu đồ phân rã chức năng cung cấp cơ sở cho việc thiết kế chi tiết và tối ưu hóa sản phẩm.

1

Xác định các chức năng chính

Xác định các chức năng cấp cao nhất mà sản phẩm hay dịch vụ cần đáp ứng.

2

Phân tích các chức năng phụ

Chia nhỏ các chức năng chính thành các chức năng con chi tiết hơn.

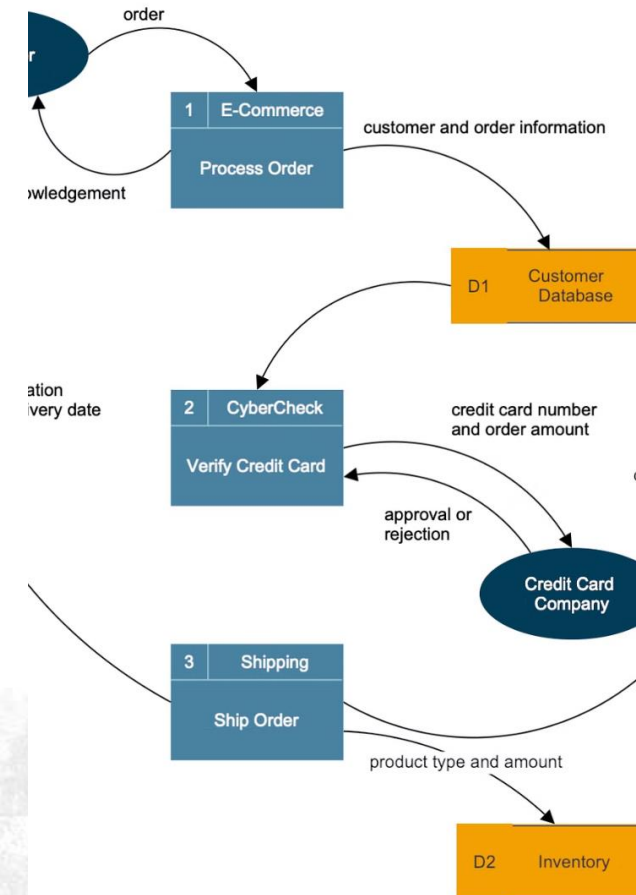
3

Liên kết các chức năng

Xác định mối quan hệ giữa giữa các chức năng và sắp xếp chúng theo một trật tự logic.

■ *Data Flow Diagram – DFD*

- Biểu đồ dòng dữ liệu là một phương pháp trực quan hóa quan trọng để hiểu rõ các luồng dữ liệu trong một hệ thống, quy trình hoặc tổ chức. Nó cho phép người xem dễ dàng nhận biết và theo dõi các thông tin, dữ liệu đang di chuyển, tương tác và biến đổi thông qua các bước trong một quy trình.



Mục đích sử dụng biểu đồ dòng dữ liệu



Phân tích quy trình

Biểu đồ dòng dữ liệu giúp doanh nghiệp hiểu rõ các bước trong quy trình hoạt động, từ đó xác định vấn đề và cơ hội cải thiện.

Tối ưu hóa quy trình

Thông qua biểu đồ, các quy trình có thể được phân tích và tối ưu hóa để tăng tăng hiệu quả, giảm thiểu lãng phí.

Truyền đạt thông tin

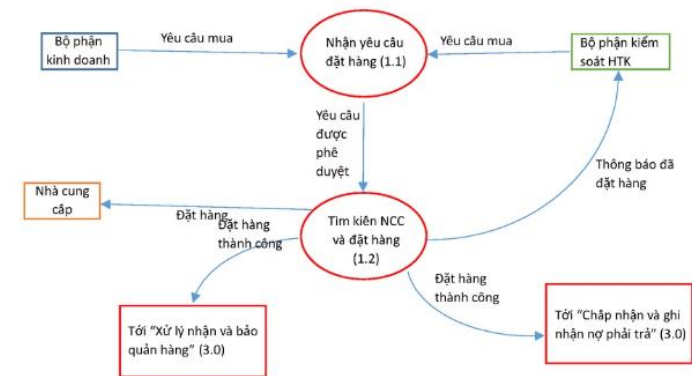
Biểu đồ dòng dữ liệu rõ ràng và dễ hiểu, giúp giao tiếp và chia sẻ thông tin giữa các bên liên quan.

Lợi ích của việc sử dụng biểu đồ dòng dữ liệu

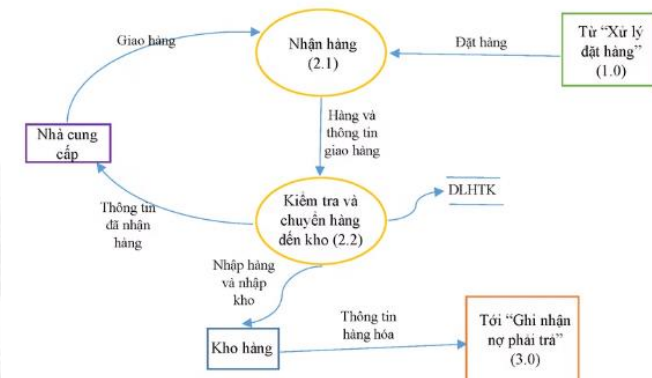
- Biểu đồ dòng dữ liệu giúp người dùng nắm bắt được quy trình và luồng dữ liệu trong một hệ thống một cách rõ ràng. Nó tăng cường tính minh bạch, giúp các bên liên quan dễ dàng theo dõi và hiểu rõ các công đoạn xử lý dữ liệu.
- Thông qua việc sử dụng biểu đồ, các tổ chức có thể phân tích hiệu quả các quy trình, xác định được các điểm nghẽn và đề ra các biện pháp cải thiện. Điều này đóng góp vào việc nâng cao năng suất và hiệu quả hoạt động.

❖ Sơ đồ dòng dữ liệu cấp 1 cho từng hoạt động

➢ Sơ đồ dòng dữ liệu cấp 1 – Xử lý đặt hàng (1.0)

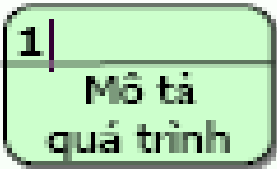

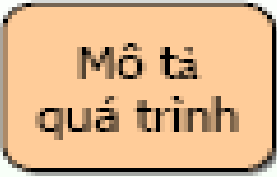
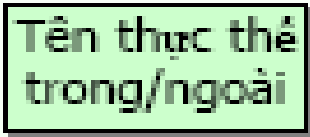
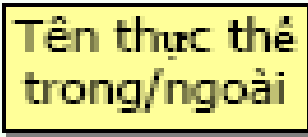

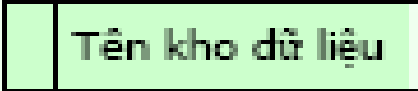
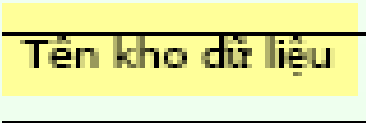


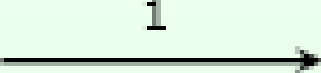


➢ Sơ đồ dòng dữ liệu cấp 1 – Nhận và bảo quản hàng (2.0)



Biểu đồ dòng dữ liệu

■ Một số phương pháp vẽ DFD

Phương pháp	Gane-Sarson	DeMarco-Yourdon	Merise
Quá trình			
Thực thể			
Kho dữ liệu			
Dòng dữ liệu			

- DFD – Các bước thực hiện
 - Phân rã chức năng hệ thống
 - Liệt kê các tác nhân, các khoản mục dữ liệu
 - Vẽ DFD cho các mức
 - Làm mịn DFD cho các mức từ DFD ngữ cảnh

1

Xác định mục đích

Trước khi bắt đầu, cần xác định rõ mục đích của việc xây dựng biểu đồ dòng dữ liệu, để có thể định hướng thiết kế cho phù hợp.

2

Định nghĩa các thành phần

Xác định các đối tượng, sự kiện, luồng dữ liệu và các mối quan hệ cần được thể hiện trong biểu đồ.

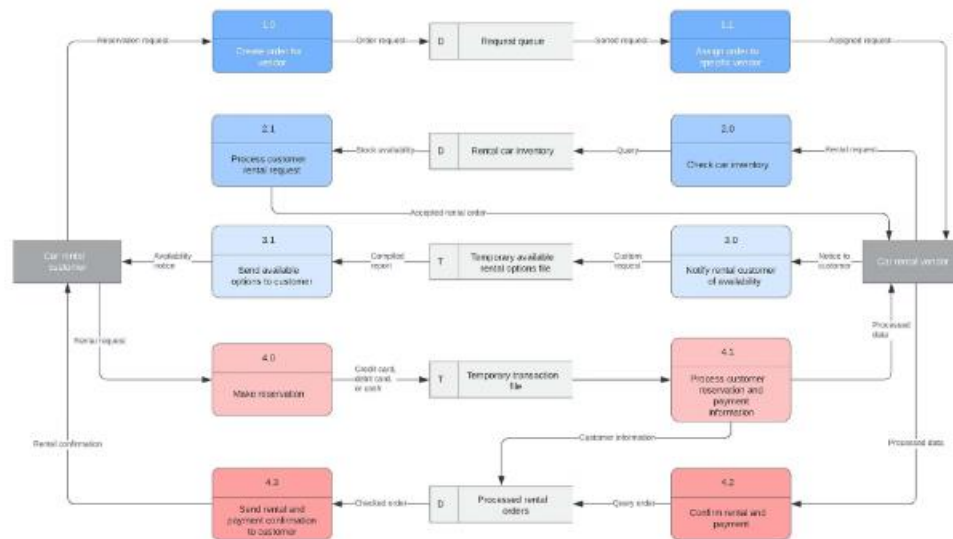
3

Thiết kế cấu trúc

Xây dựng sơ đồ khái quát về luồng dữ liệu và các thành phần, để làm cơ sở cho việc triển khai chi tiết.

Ví dụ về ứng dụng biểu đồ dòng dữ liệu

- Biểu đồ dòng dữ liệu được ứng dụng rộng rãi trong nhiều lĩnh vực, từ phân tích quy trình kinh doanh, theo dõi chuỗi cung ứng, đến quản lý dự án và phát triển phần mềm.
- Ví dụ, trong lĩnh vực phát triển phần mềm, biểu đồ dòng dữ liệu được sử dụng để mô tả luồng thông tin và quy trình xử lý dữ liệu trong hệ thống, giúp các nhà phát triển hiểu rõ các thành phần và mối quan hệ giữa chúng.



1 Phức tạp về dữ liệu

Các dòng dữ liệu phức tạp với nhiều biến số và quan hệ có thể khiến việc xây dựng biểu đồ trở nên cồng kềnh và khó hiểu.

2 Khó đạt được sự rõ ràng

Việc truyền tải thông tin một cách đơn giản, rõ ràng trong biểu đồ dòng dữ liệu là thách thức, đặc biệt khi có nhiều chi tiết và các quan hệ phức tạp.

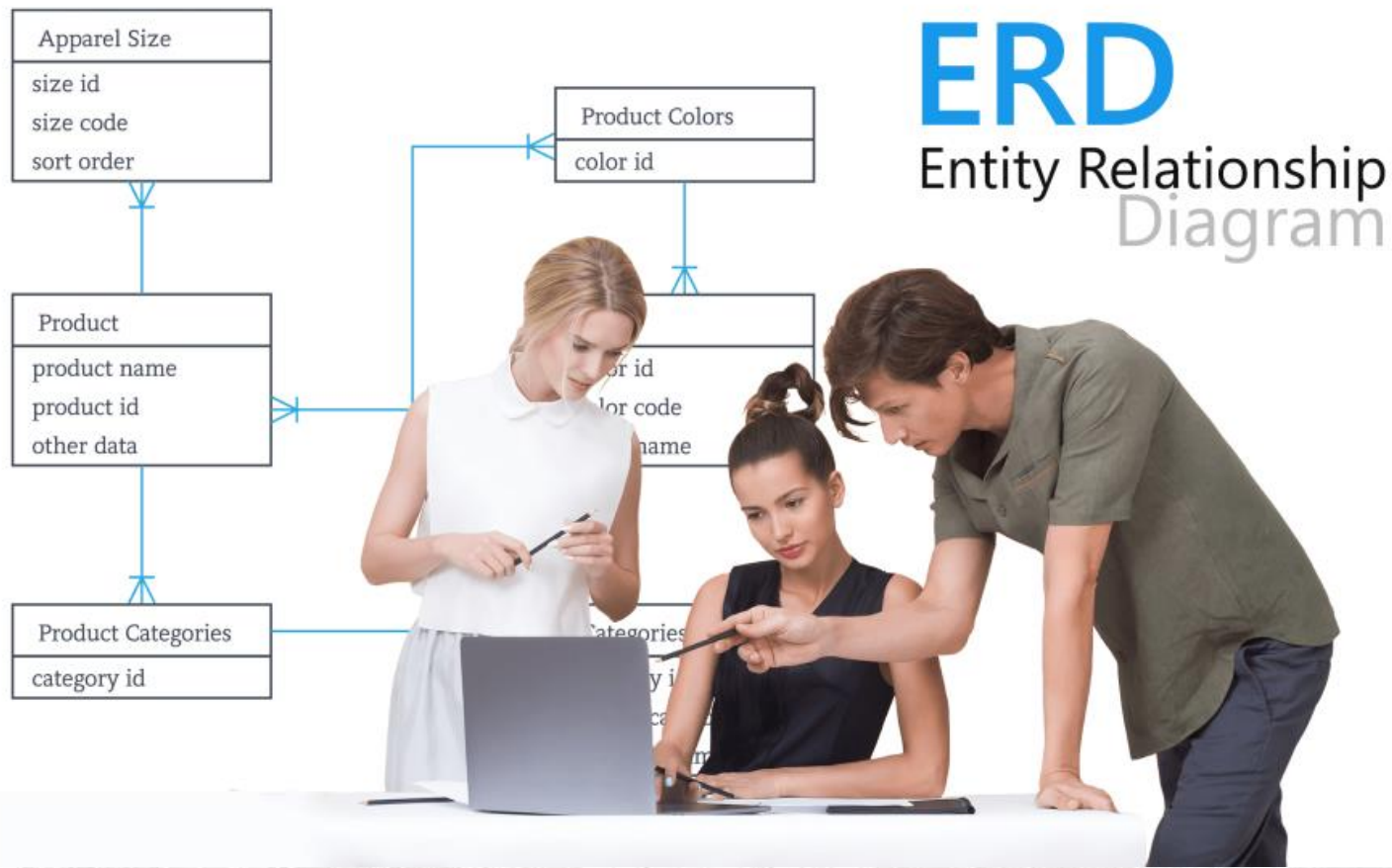
3 Quá nhiều thông tin

Biểu đồ dòng dữ liệu có thể bao gồm quá nhiều chi tiết, làm cho họ trở nên rối mắt và khó theo dõi.

4 Giải pháp: Tối giản hóa và trực quan hóa

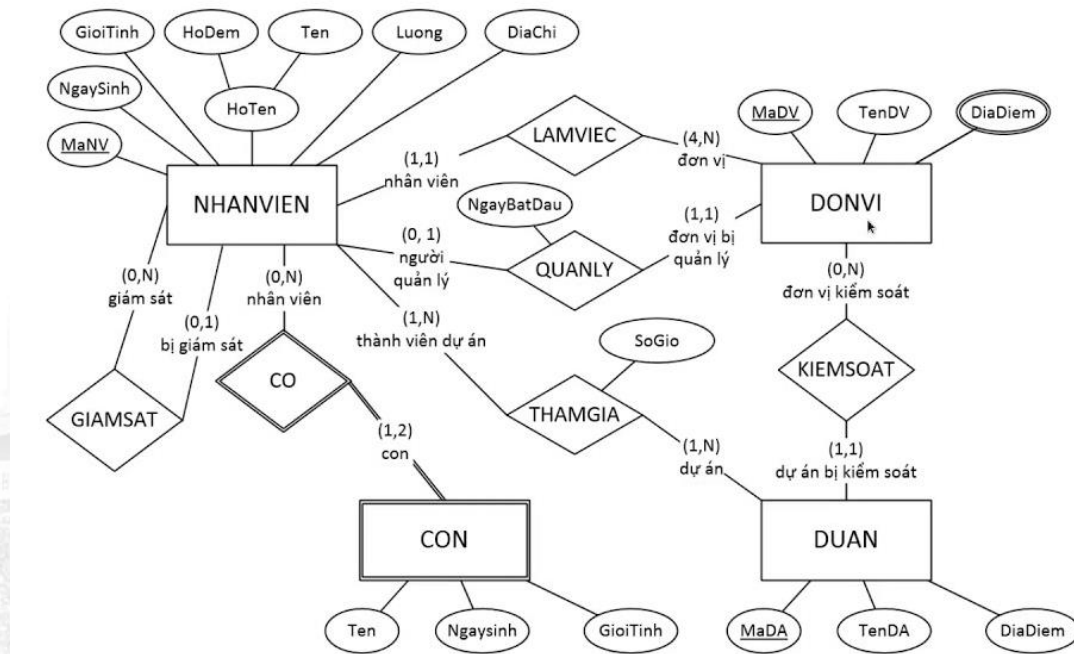
Sử dụng các biểu tượng, mã màu và bố cục hợp lý để đơn giản hóa biểu đồ, đồng thời tập trung vào các thông tin then chốt.

ERD - Sơ đồ quan hệ thực thể



Sơ đồ quan hệ thực thể

- Sơ đồ quan hệ thực thể (Entity Relationship Diagram - ERD) là một công cụ quan trọng trong việc thiết kế và phát triển cơ sở dữ liệu. Nó cung cấp một cái nhìn trực quan về các thực thể và mối quan hệ giữa chúng, giúp đơn giản hóa việc hiểu và quản lý dữ liệu trong hệ thống.



Các thành phần cơ bản của sơ đồ quan hệ thực thể



Thực thể

Là đối tượng trong thế giới thực, có thể là sự vật, sự kiện hoặc khái niệm cần được lưu trữ và quản lý trong hệ thống.

Thuộc tính

Là những đặc trưng hay đặc điểm mô tả các thực thể, giúp phân biệt các thực thể với nhau.

Quan hệ

Là mối liên kết giữa các thực thể, giúp mô tả các tương tác và liên hệ trong hệ thống.

Khóa

Là những thuộc tính hoặc tập hợp các thuộc tính duy nhất định danh từng thực thể.

- Quan hệ **một-nhiều (1:M)**: Thể hiện mối quan hệ giữa một thực thể và nhiều thực thể khác, ví dụ như quan hệ giữa Khách hàng và Đơn đặt hàng.
- Quan hệ **một-một (1:1)**: Thể hiện mối quan hệ giữa hai thực thể, mỗi thực thể chỉ có thể liên kết với tối đa một thực thể khác, ví dụ như quan hệ giữa Nhân viên và Hồ sơ nhân viên.
- Quan hệ **nhiều-nhiều (M:N)**: Thể hiện mối quan hệ giữa nhiều thực thể với nhiều thực thể khác, ví dụ như quan hệ giữa Sách và Độc giả.

Quy tắc vẽ sơ đồ quan hệ thực thể

Xác định các thực thể

Đầu tiên, cần xác định các thực thể chính trong hệ thống. Các thực thể này đại diện cho các đối tượng, sự kiện hoặc khái niệm quan trọng.

1

2

3

Xác định các mối quan hệ

Sau đó, cần xác định các mối quan hệ giữa các thực thể. Các mối quan hệ này có thể là một-một, một-nhiều hoặc nhiều-nhiều.

Mô tả các thuộc tính

Tiếp theo, cần mô tả các thuộc tính của từng thực thể. Các thuộc tính này cung cấp thông tin chi tiết về mỗi thực thể.

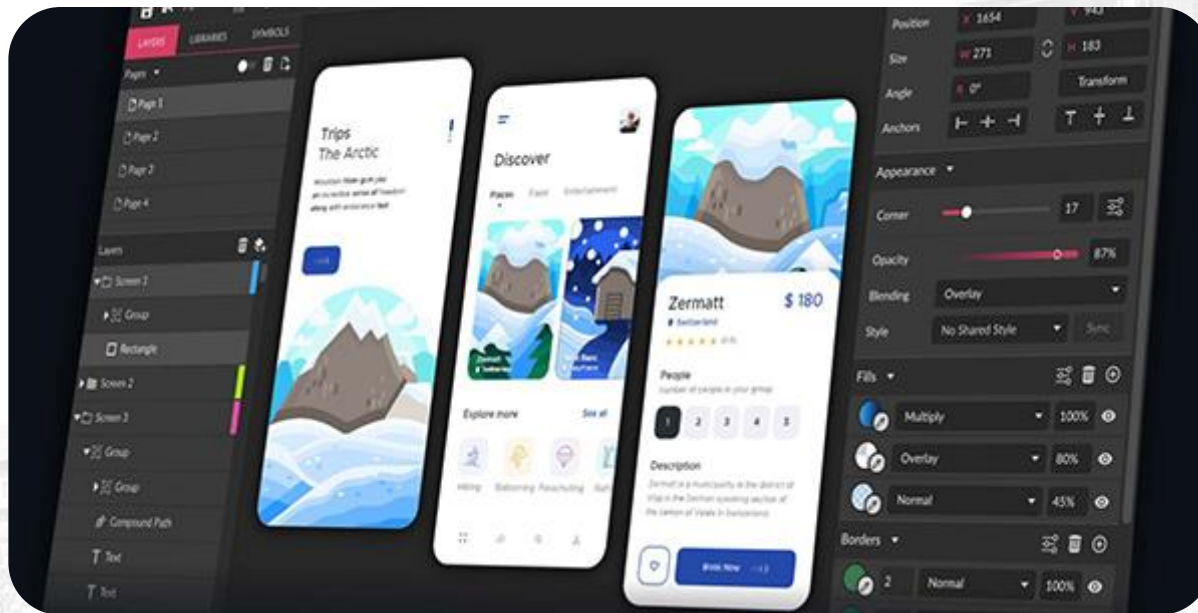
Lợi ích của sơ đồ quan hệ thực thể



- Sơ đồ quan hệ thực thể mang lại nhiều lợi ích quan trọng trong thiết kế và quản lý cơ sở dữ liệu. Nó giúp phân tích và mô hình hóa dữ liệu một cách rõ ràng, chính xác, để tổ chức dữ liệu hiệu quả và đáp ứng nhu cầu người dùng.
- Từ đó, giúp phát triển hệ thống cơ sở dữ liệu an toàn, bền vững và dễ dàng bảo trì, nâng cấp. Sơ đồ còn hỗ trợ tối ưu hóa hiệu suất, tính nhất quán và tính toàn vẹn dữ liệu trong ứng dụng.

Thiết kế bản mẫu phần mềm

- Bản mẫu phần mềm là mô hình hóa trước khi phát triển phần mềm chính thức. Nó giúp các nhà phát triển có cái nhìn tổng quan về giao diện, chức năng và luồng hoạt động của phần mềm. Quá trình thiết kế bản mẫu là rất quan trọng để đảm bảo phần mềm đáp ứng nhu cầu của người dùng.

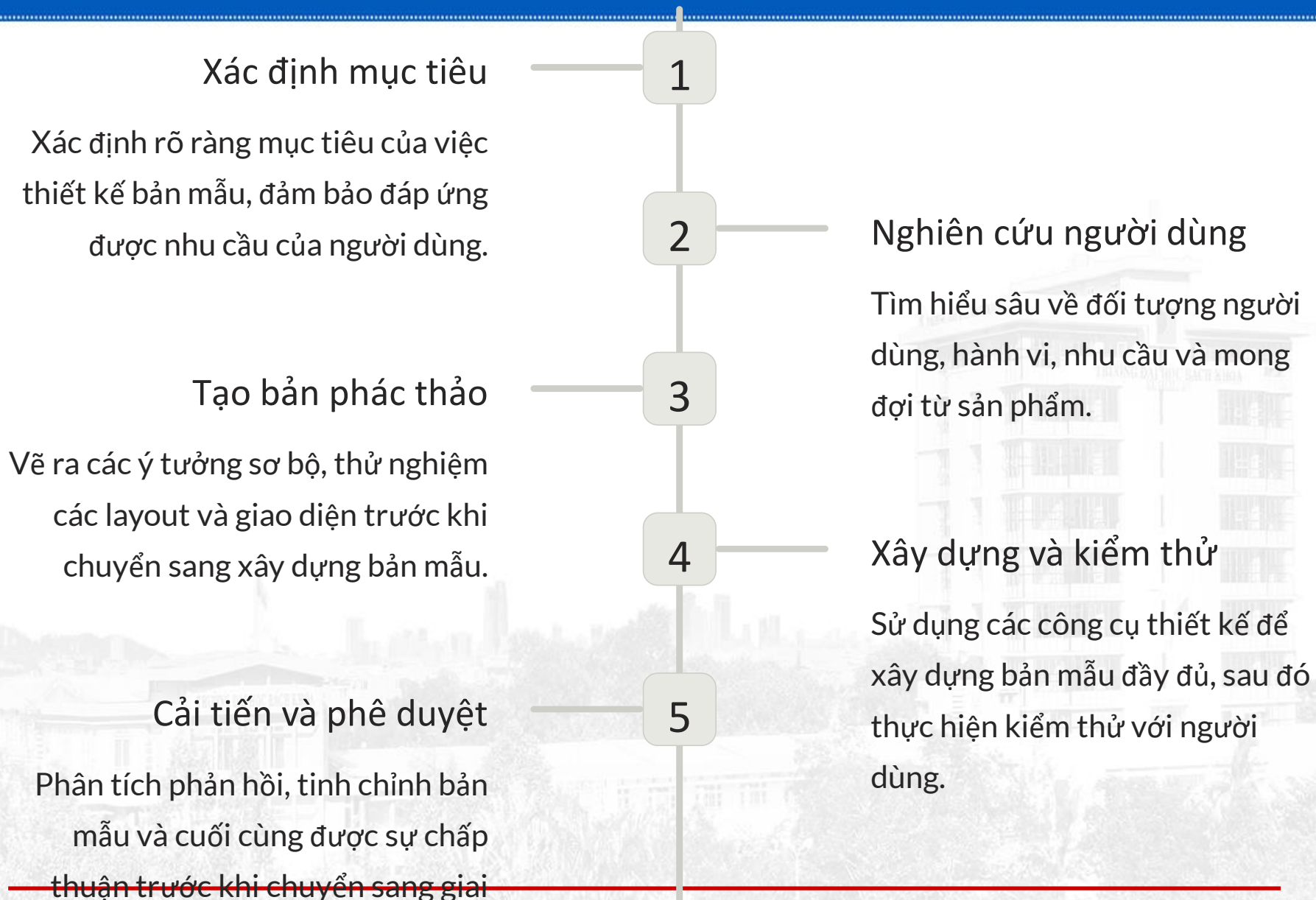


Vai trò và tầm quan trọng của thiết kế bản mẫu

- Thiết kế bản mẫu phần mềm đóng vai trò quan trọng trong quá trình phát triển sản phẩm. Nó giúp xác định và nghiên cứu các yêu cầu của người dùng, cải thiện trải nghiệm người dùng, và tối ưu hóa quy trình làm việc. Bản mẫu cũng cho phép kiểm tra và đánh giá các ý tưởng trước khi chính thức triển khai.
- Việc áp dụng thiết kế bản mẫu mang lại nhiều lợi ích như giảm chi phí, rút ngắn thời gian phát triển, và tăng tính linh hoạt. Nó cho phép nhóm phát triển phản hồi nhanh chóng với nhu cầu của người dùng, đồng thời giảm rủi ro gặp phải các vấn đề nghiêm trọng trong giai đoạn cuối.

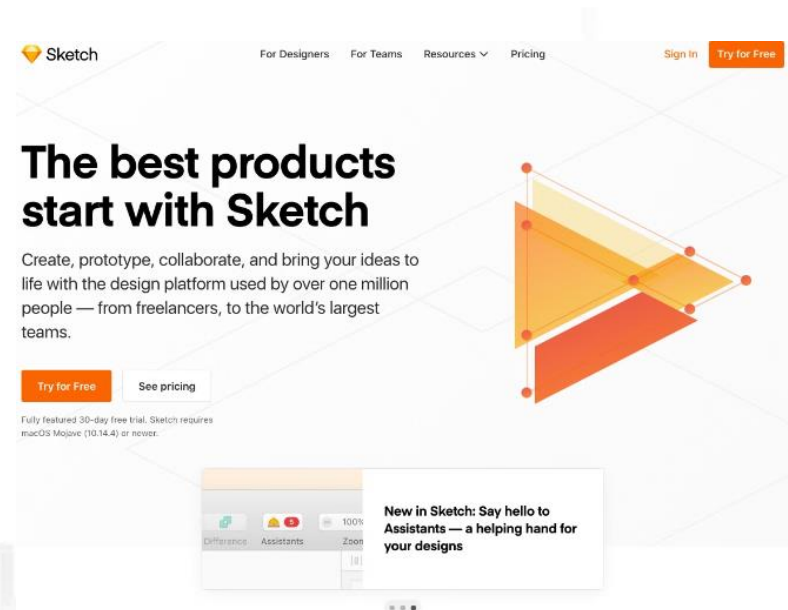


Các bước trong quá trình thiết kế bản mẫu



Tạo ra các bản nháp và phác thảo sơ bộ

- Sau khi xác định mục tiêu và yêu cầu của người dùng, bước tiếp theo là tạo ra các bản nháp và phác thảo sơ bộ. Đây là giai đoạn quan trọng để thử nghiệm và khám phá các ý tưởng trước khi chính thức xây dựng các bản mẫu.
- Các bản nháp và phác thảo sơ bộ có thể bao gồm các kịch bản sử dụng, wireframe, bố cục của giao diện người dùng, các concept thiết kế và các ý tưởng khác. Mục đích là tạo ra các phiên bản sơ khai, dễ điều chỉnh và cải tiến để tìm ra phương án tối ưu.



Xây dựng và kiểm tra các bản mẫu



Xây dựng bản mẫu

Dựa trên các đề xuất và ý tưởng từ quá trình thiết kế, các bản mẫu cần được xây dựng một cách chi tiết và chính xác. Các bản mẫu này sẽ được sử dụng để kiểm tra và đánh giá trước khi triển khai chính thức.

Kiểm tra các bản mẫu

Các bản mẫu sẽ được kiểm tra kỹ lưỡng bởi nhóm sản phẩm và người dùng mục tiêu. Họ sẽ đánh giá về tính direct sử dụng, trải nghiệm người dùng và mức độ đáp ứng các yêu cầu ban đầu.

Tinh chỉnh và cải tiến

Dựa trên phản hồi của kiểm tra, các bản mẫu sẽ tiếp tục được tinh chỉnh và cải tiến cho đến khi đáp ứng được tất cả các yêu cầu. Quá trình này giúp đảm bảo sản phẩm hoàn thiện và sẵn sàng triển khai.

Đánh giá và cải thiện các bản mẫu



1

Đánh giá hiệu quả của bản mẫu

Kiểm tra xem bản mẫu có đáp ứng được nhu cầu và yêu cầu của người dùng không. Lắng nghe phản hồi và đề xuất để cải thiện.

2

Cập nhật và điều chỉnh bản mẫu

Dựa trên đánh giá, cập nhật và điều chỉnh bản mẫu để tối ưu hóa trải nghiệm người dùng.

3

Kiểm tra lại và đánh giá lại

Kiểm tra lại bản mẫu đã cải thiện và đánh giá xem có đáp ứng được yêu cầu không. Lặp lại quá trình điều chỉnh nếu cần thiết.

4

Tích hợp bản mẫu vào sản phẩm

Sau khi hoàn thiện, tích hợp bản mẫu vào sản phẩm phần mềm để đảm bảo trải nghiệm người dùng tốt.

Thiết kế chuẩn hóa

Bản mẫu được tích hợp chặt chẽ vào quy trình phát triển, đảm bảo thiết kế phần mềm tuân thủ các tiêu chuẩn và quy chuẩn.

Phối hợp nhóm

Nhóm thiết kế, phát triển và kiểm thử cùng làm việc chặt chẽ để đảm bảo bản mẫu được triển khai hiệu quả.

Phản hồi và cải tiến

Bản mẫu được liên tục cải tiến dựa trên phản hồi từ người dùng và các vòng lặp phát triển.

Giám sát chất lượng

Các bản mẫu được đánh giá và kiểm tra chặt chẽ để đảm bảo tính nhất quán và đáp ứng yêu cầu của người dùng.

Kết luận và lợi ích của thiết kế bản mẫu



Gia tăng trải nghiệm người dùng

Thiết kế bản mẫu giúp cải thiện giao diện người dùng, tối ưu hóa quy trình và nâng cao trải nghiệm sử dụng của ứng dụng. Điều này tăng khả năng thu hút và giữ chân người dùng.

Phát triển sản phẩm hiệu quả

Quy trình thiết kế bản mẫu giúp xác định rõ yêu cầu và tính năng cần ưu tiên, từ đó định hướng phát triển sản phẩm hiệu quả, đáp ứng nhu cầu thực tế của người dùng.

Giảm rủi ro và chi phí

Bằng cách phát hiện và sửa chữa lỗi sớm trong giai đoạn thiết kế, bản mẫu giúp giảm thiểu rủi ro và chi phí phát triển sau này.

Tăng tính cạnh tranh

Sản phẩm có thiết kế bản mẫu chuyên nghiệp sẽ nâng cao tính thẩm mỹ và trải nghiệm người dùng, giúp sản phẩm trở nên nổi bật và cạnh tranh hơn trên thị trường.

- Máy trạng thái hữu hạn (FSM) là một mô hình toán học đơn giản để mô tả trạng thái và quy luật chuyển đổi của một hệ thống. Nó được sử dụng rộng rãi trong lập trình, thiết kế phần cứng, và nhiều ứng dụng khác.

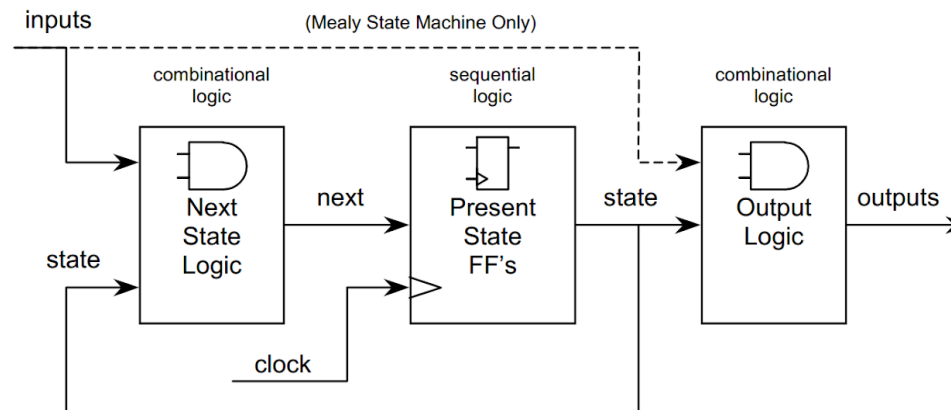


Figure 1 - FSM Block Diagram

■ Khái niệm

- Máy trạng thái hữu hạn (Finite State Machine) là một mô hình toán học dùng để mô tả hành vi của một hệ thống với các trạng thái nhất định và các sự kiện xảy ra để chuyển đổi từ trạng thái này sang trạng thái khác.

■ Lợi ích

- FSM cho phép mô hình hóa các hệ thống phức tạp một cách dễ hiểu, dễ cài đặt và dễ bảo trì. Nó là cơ sở của nhiều ứng dụng công nghệ như điều khiển tự động, xử lý ngôn ngữ tự nhiên và lập trình máy tính.

■ Finite State Machine

- Mô tả các luồng điều khiển
- Biểu diễn dạng đồ thị
- Bao gồm:
 - Tập hợp các trạng thái S (các nút của đồ thị)
 - Tập hợp các dữ liệu vào I (các nhãn của các cung)
 - Tập hợp các chuyển tiếp $T: S \times I \rightarrow S$ (các cung có hướng của đồ thị)
 - Khi có một dữ liệu vào một trạng thái sẽ chuyển sang trạng thái khác

Ứng dụng của FSM

Điều khiển hệ thống

1

FSM được sử dụng rộng rãi trong việc kiểm soát và điều khiển các hệ thống phức tạp như máy tính, thiết bị điện tử, robot và các hệ thống tự động khác.

Mô hình hóa giao thức mạng

3

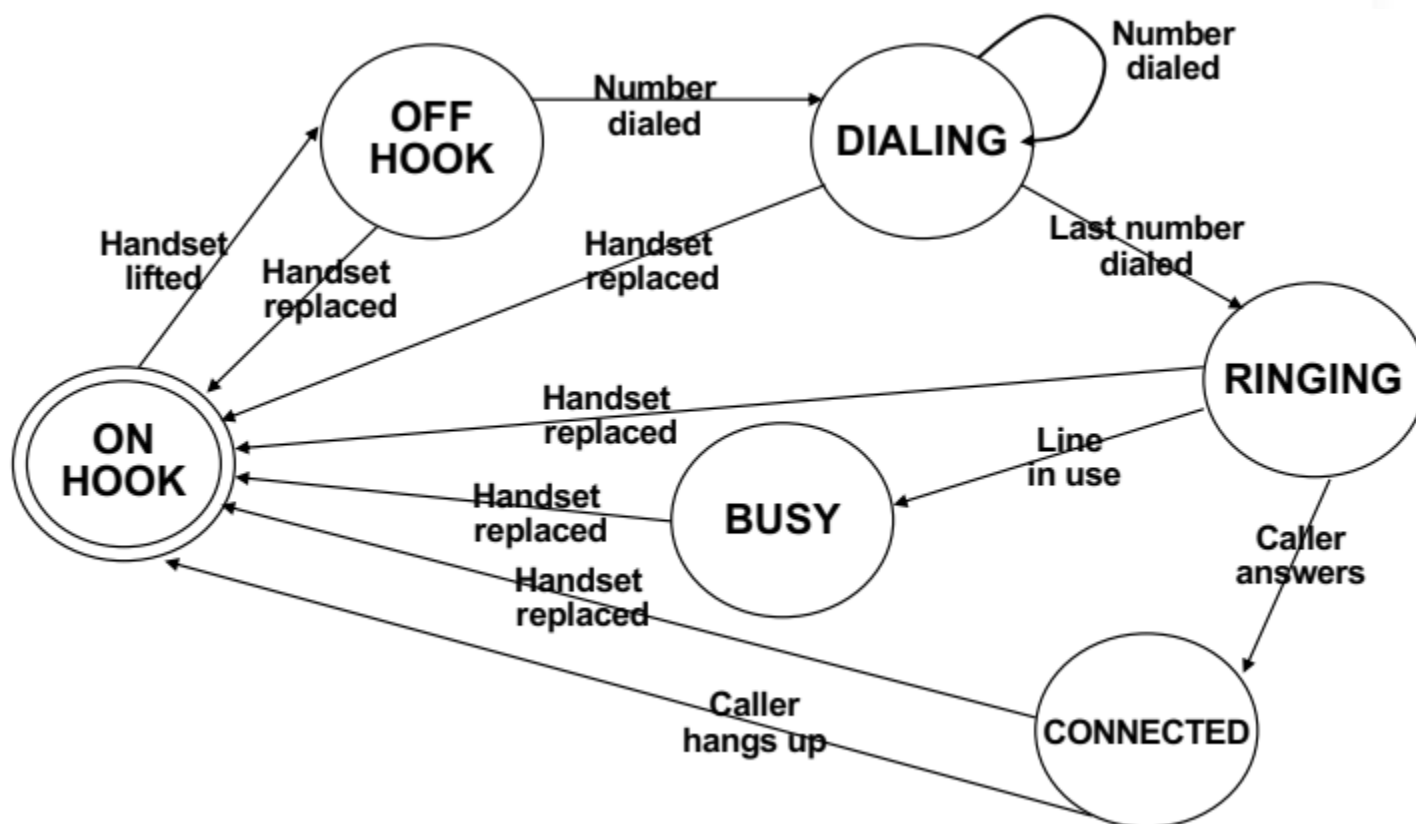
FSM được sử dụng để mô hình hóa và mô tả các giao thức mạng như TCP/IP, HTTP và Bluetooth, giúp tối ưu hóa hiệu suất và độ tin cậy của các giao thức này.

Xử lý ngôn ngữ

FSM là nền tảng cho các ứng dụng xử lý ngôn ngữ tự nhiên như trình điều khiển trò chuyện, bộ phân tích cú pháp và các công cụ văn bản tự động.

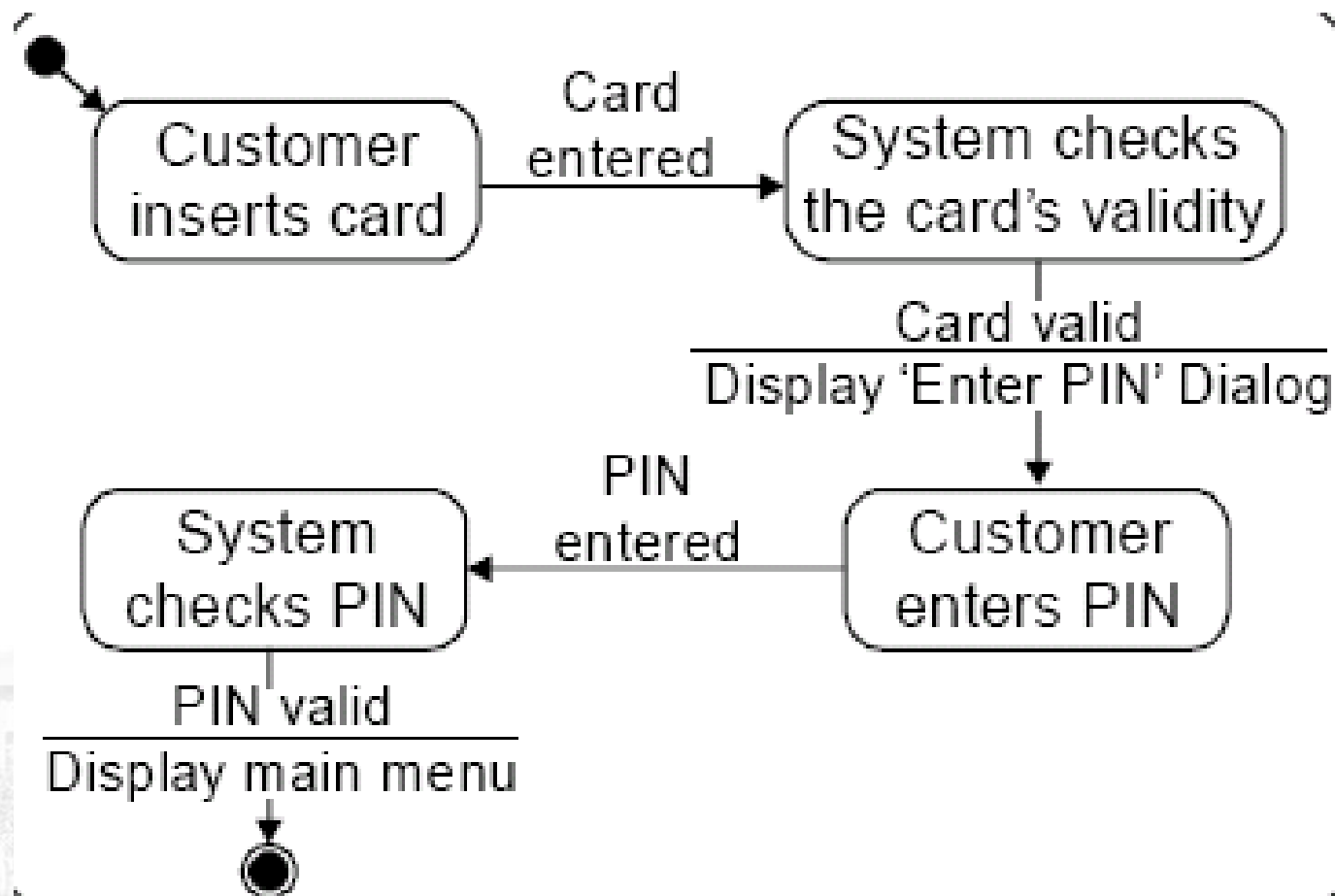
Máy trạng thái hữu hạn

■ Ví dụ 1:



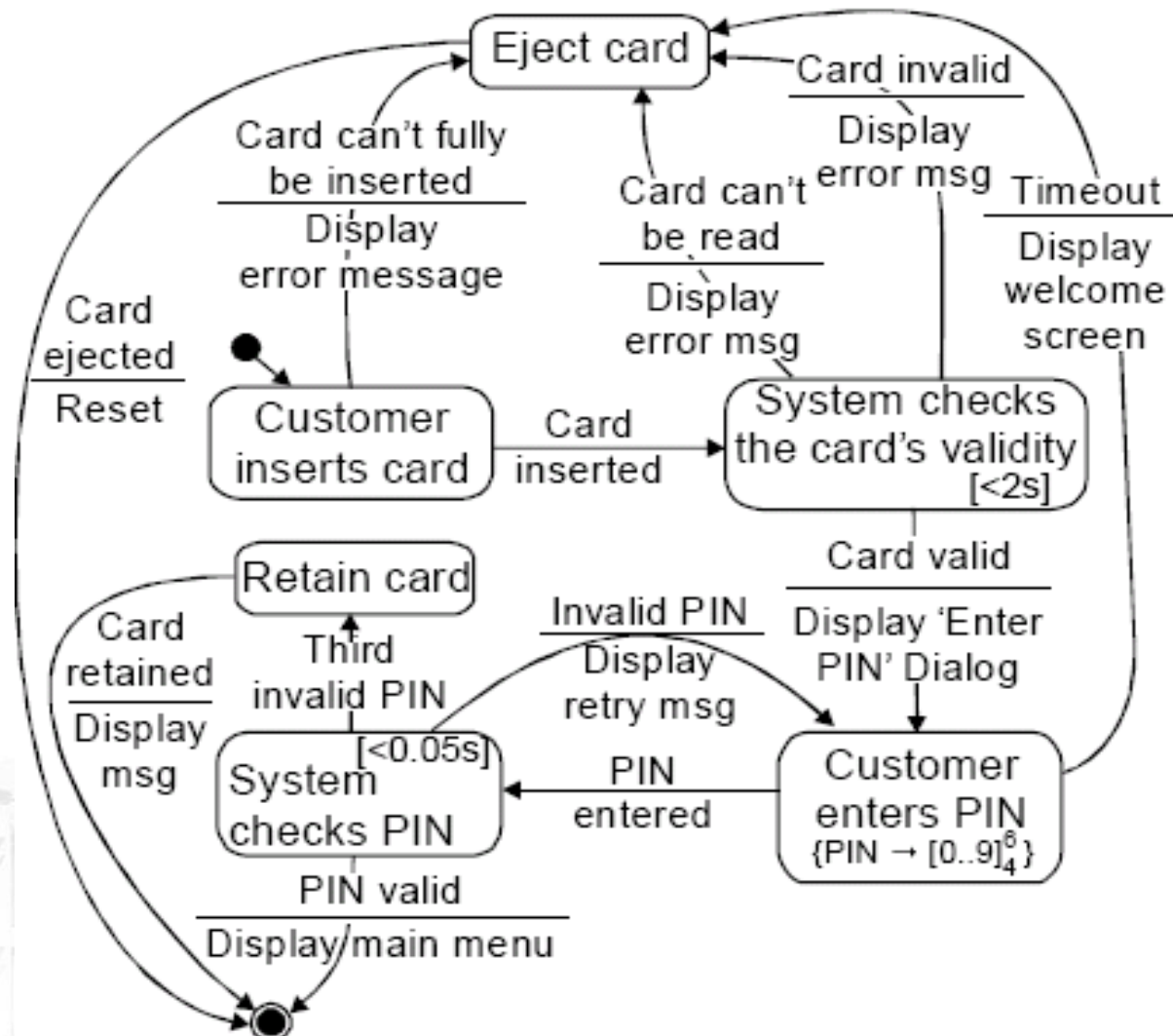
Máy trạng thái hữu hạn

■ Ví dụ 2:



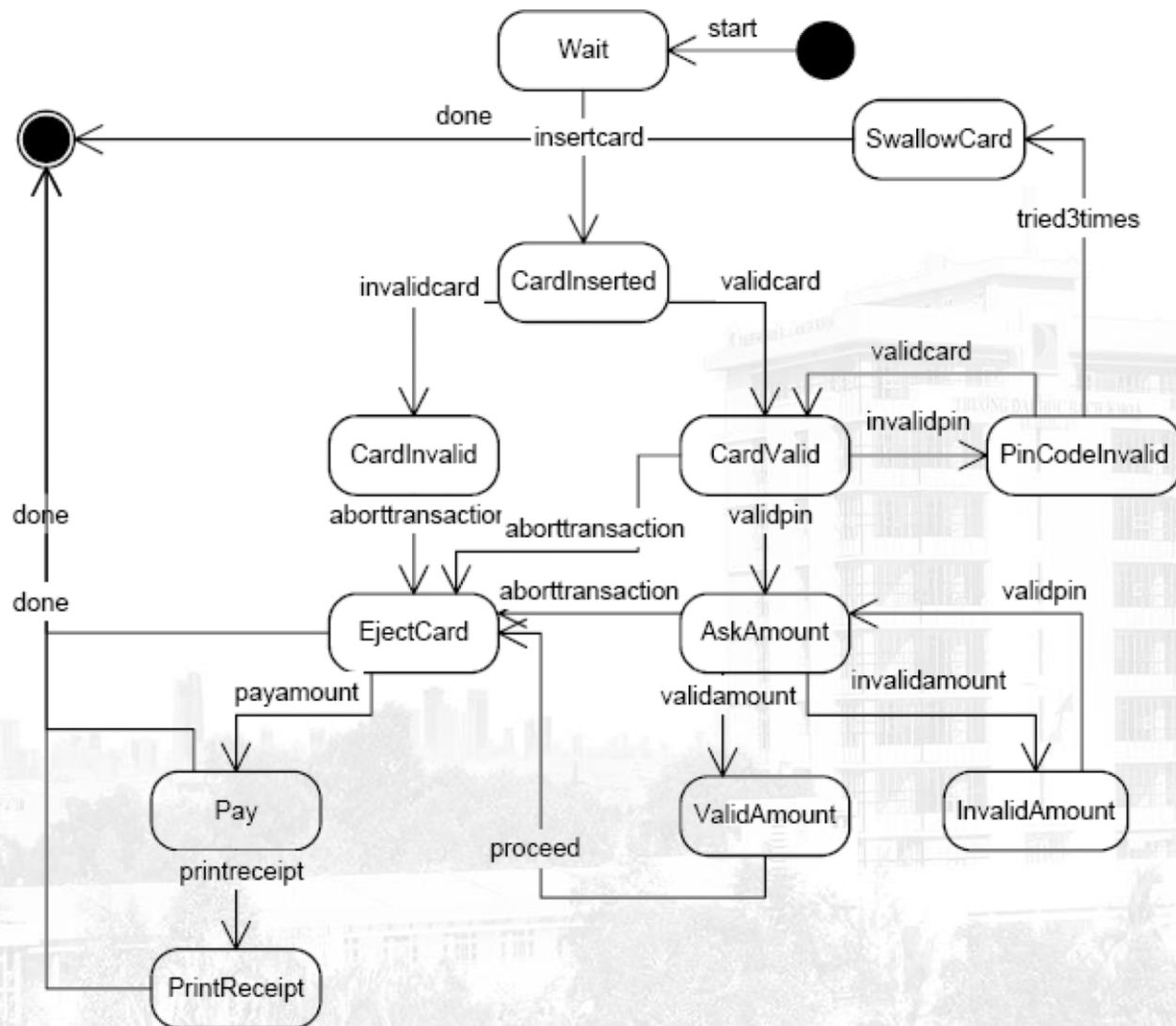
Máy trạng thái hữu hạn

■ Ví dụ 2:



Máy trạng thái hữu hạn

■ Ví dụ 2:



■ Ưu điểm

- Đơn giản, dễ hiểu và dễ triển khai. Cho phép mô hình hóa các hệ thống phức tạp một cách trực quan.

■ Dễ dàng mở rộng và nâng cấp

- Các trạng thái và chuyển tiếp có thể dễ dàng được thêm vào hoặc chỉnh sửa để đáp ứng các yêu cầu mới.

■ Hiệu quả về tài nguyên

- FSM không yêu cầu nhiều tài nguyên hệ thống và có thể được triển khai trên các nền tảng phần cứng khác nhau.

■ Nhược điểm

- Khó mô hình hóa các hệ thống phức tạp với nhiều trạng thái và chuyển tiếp. Có thể trở nên cồng kềnh và khó quản lý.



Công cụ mô phỏng

Các phần mềm như Proteus, Logisim, Multisim giúp mô hình hóa và mô phỏng FSM trực quan.

01
10

Ngôn ngữ lập trình

FSM có thể được mã hóa bằng các ngôn ngữ lập trình như C, Java, Python, Verilog và VHDL.



Thiết kế phần cứng

Các công cụ thiết kế phần cứng như FPGA và ASIC được sử dụng để triển khai FSM trên phần cứng.



Biểu đồ trạng thái

Biểu đồ trạng thái (state diagram) được dùng để mô tả và phân tích FSM trực quan.

- Trong tương lai, máy trạng thái hữu hạn (FSM) sẽ tiếp tục phát triển và mở rộng phạm vi ứng dụng. Các xu hướng chính bao gồm tích hợp với công nghệ AI, mở rộng sang lĩnh vực xử lý ngôn ngữ tự nhiên và ứng dụng trong các hệ thống phân tán, IoT.
- Các nhà nghiên cứu cũng đang nghiên cứu cách mở rộng FSM để xử lý các tình huống phức tạp hơn, như FSM không xác định, FSM có thể tự tạo trạng thái mới và FSM phân tán. Điều này sẽ làm tăng khả năng ứng dụng của FSM trong các hệ thống thông minh và tự động.

Giới thiệu về Mạng Petri



- Mạng Petri là một công cụ mô hình hóa và phân tích các hệ thống phân tán và song song. Nó đóng vai trò quan trọng trong việc thiết kế và phân tích các hệ thống công nghệ thông tin, công nghiệp và quá trình kinh doanh.

Định nghĩa và ứng dụng



- Mạng Petri là một mô hình toán học dùng để mô tả sự phân phối, đồng thời và tương tác của các nguồn lực. Nó có thể được sử dụng để mô hình hóa, phân tích và kiểm soát các hệ thống phân tán, song song, không xác định và không đồng bộ.
- Mạng Petri có ứng dụng rộng rãi trong nhiều lĩnh vực như công nghệ, giao thông, logistic, y tế, và nhiều lĩnh vực khác. Nó đặc biệt hiệu quả trong việc phân tích và thiết kế các hệ thống phức tạp.
- Với khả năng mô hình hóa các hệ thống động và logic, Mạng Petri được ứng dụng rộng rãi trong các lĩnh vực như kiểm soát sản xuất, giao thông, quản lý dự án, và nhiều ứng dụng khác.

Mạng Petri (Petri Nets)



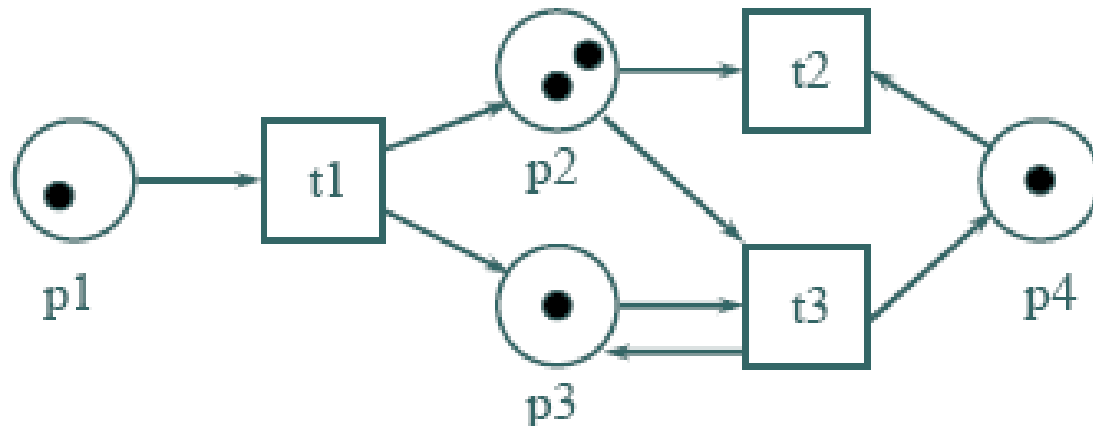
- Thích hợp để mô tả các hệ thống không đồng bộ với những hoạt động đồng thời
- Mô tả luồng điều khiển của hệ thống
- Đề xuất năm 1962 bởi Carl Adam
- Có loại
 - Mạng Petri (cổ điển)
 - Mạng Petri mở rộng

Mạng Petri (Petri Nets)

- Gồm các phần tử
 - Một tập hữu hạn các **nút** (O)
 - Một tập hữu hạn các **chuyển tiếp** (\square)
 - Một tập hữu hạn các **cung** (\rightarrow)
 - Các cung nối các nút với các chuyển tiếp hoặc ngược lại
 - Mỗi nút có thể chứa một hoặc nhiều **thẻ** (\bullet)

Mạng Petri (Petri Nets)

■ Ví dụ:

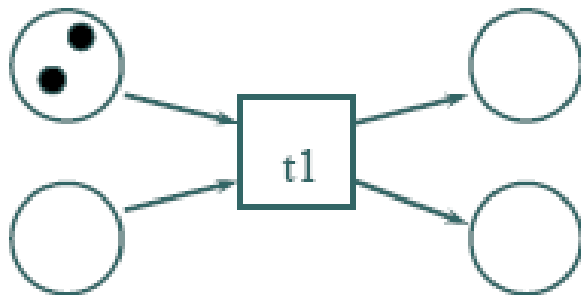


Mạng Petri (Petri Nets)

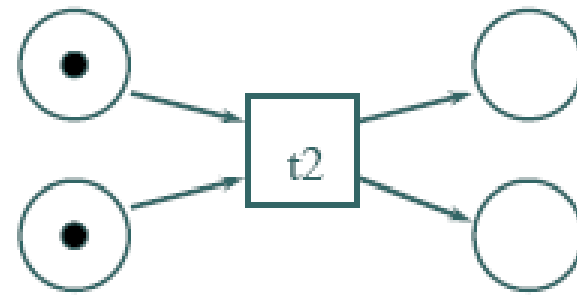
- Mạng Petri được định nghĩa bởi sự đánh dấu các nút của nó
- Việc đánh dấu các nút được tiến hành theo nguyên tắc sau:
 - Mỗi chuyển tiếp có các nút vào và các nút ra
 - Nếu tất cả các nút vào của một chuyển tiếp có ít nhất một thẻ, thì chuyển tiếp này có thể vượt qua được.
 - Nếu chuyển tiếp này được thực hiện thì tất cả các nút vào của chuyển tiếp sẽ bị lấy đi một thẻ, và một thẻ sẽ được thêm vào tất cả các nút ra của chuyển tiếp
 - Nếu chuyển tiếp là có thể vượt qua thì chọn chuyển tiếp nào cũng được

Mạng Petri (Petri Nets)

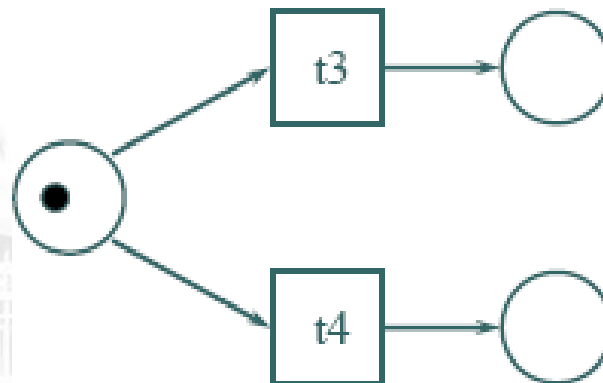
■ Ví dụ



t1 không thể vượt qua được



t2 có thể vượt qua được



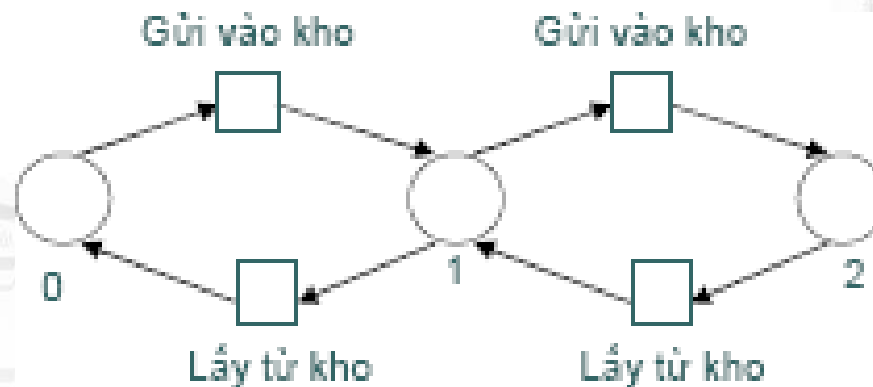
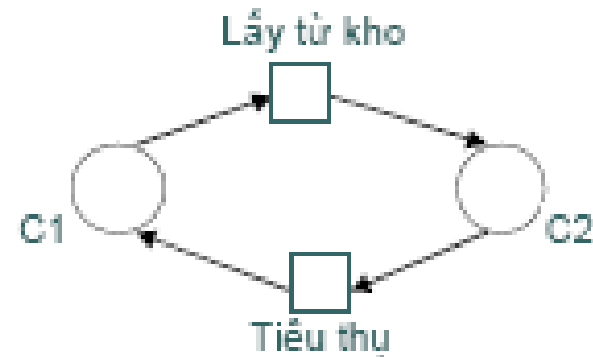
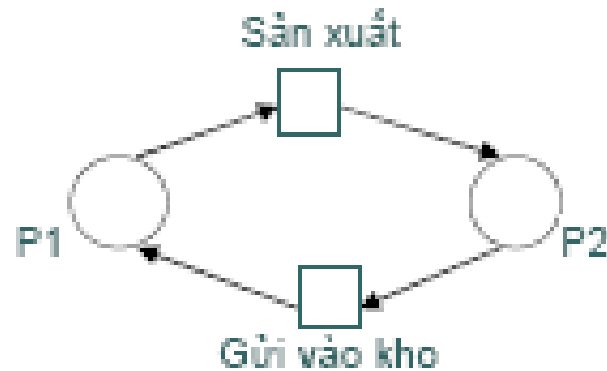
hoặc t3 được vượt qua
hoặc t4 được vượt qua

■ Ví dụ 4:

- Hệ thống cần mô tả bao gồm một nhà sản xuất, một nhà tiêu thụ và một kho hàng chỉ chứa được nhiều nhất hai sản phẩm
- Nhà sản xuất có 2 trạng thái:
 - P1: không sản xuất
 - P2 đang sản xuất
- Nhà tiêu thụ có 2 trạng thái:
 - C2: có sản phẩm để tiêu thụ
 - C1: không có sản phẩm để tiêu thụ
- Kho hàng có 3 trạng thái
 - Chứa 0 sản phẩm
 - Chứa 1 sản phẩm
 - Chứa 2 sản phẩm

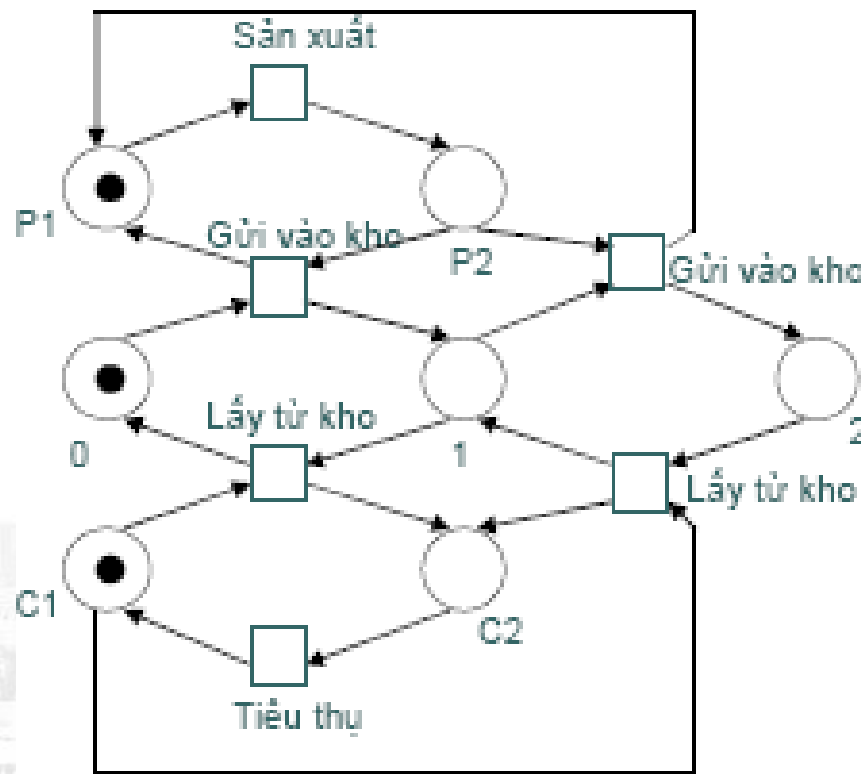
Mạng Petri (Petri Nets)

- Ví dụ 4: mô tả tách rời mỗi thành phần



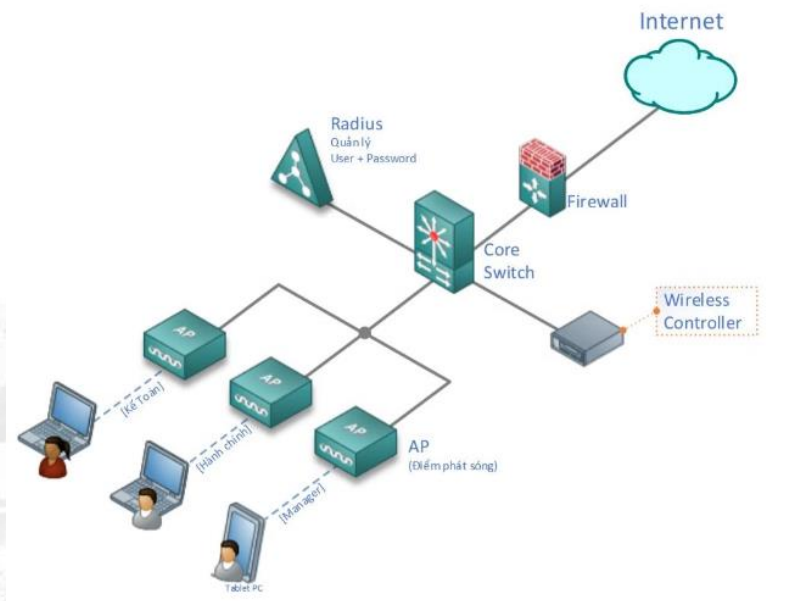
Mạng Petri (Petri Nets)

- Ví dụ 4: Mô tả kết hợp các thành phần



Phân tích và mô hình hóa bằng Mạng Petri

- Mạng Petri là một công cụ mạnh mẽ để phân tích và mô hình hóa các hệ thống phức tạp. Nó cho phép mô tả các quá trình song song, các sự kiện và các điều kiện trước khi xảy ra các sự kiện.
- Việc sử dụng Mạng Petri giúp các nhà phát triển có thể xác định và kiểm soát các tình huống bất thường, tránh được các lỗi và đảm bảo tính chính xác của hệ thống.



Lập trình và mô phỏng Mạng Petri



01
10



Lập trình Mạng Petri

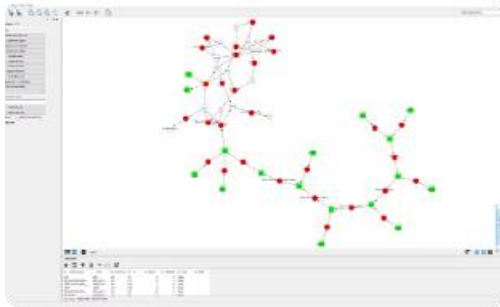
Mạng Petri có thể được lập trình bằng các ngôn ngữ cụ thể như PNML (Petri Net Markup Language) và được mô phỏng bằng các công cụ chuyên dụng như CPN Tools, PIPE, và Tina.

Mô phỏng Mạng Petri

Việc mô phỏng Mạng Petri giúp kiểm tra và phân tích các hệ thống phức tạp, cũng như dự đoán hành vi của chúng. Các công cụ mô phỏng cho phép quan sát trạng thái động của Mạng Petri và kiểm tra tính chất như khả năng đạt đến trạng thái ổn định.

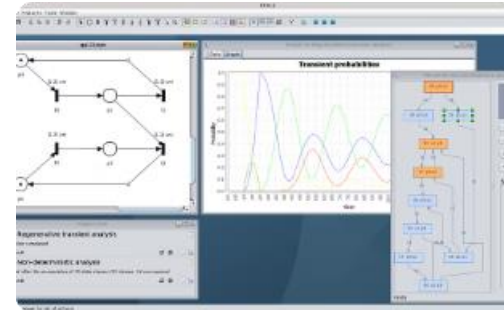
Phân tích Mạng Petri

Ngoài việc mô phỏng, Mạng Petri còn được sử dụng để phân tích các hệ thống phức tạp, bao gồm cả việc kiểm tra tính liveability, tính an toàn, tính phân hoạch, và các đặc tính khác.



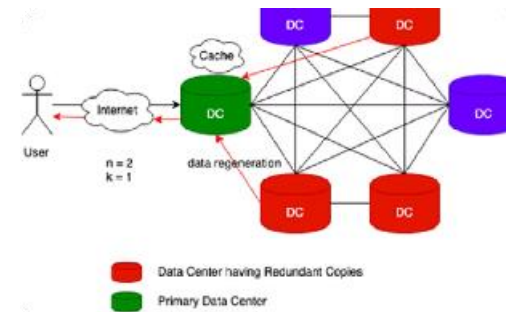
Phần mềm mô hình hóa

Các phần mềm như CPN Tools, PIPE và Renew cho phép thiết kế, phân tích và mô phỏng các mạng Petri một cách trực quan và hiệu quả.



Công cụ phân tích

Các công cụ phân tích như bảng phạm trù, đồ thị khả thi và các thuật toán kiểm tra tính chất giúp người dùng đánh giá và xác nhận các đặc tính của mạng Petri.



Tích hợp với các công cụ khác

Mạng Petri còn có thể được tích hợp với các công cụ khác như UML, MATLAB và SQL để phục vụ các yêu cầu mô hình hóa và phân tích phức tạp hơn.

- <https://petri-net-visualizer.netlify.app/>



So sánh các kỹ thuật đặc tả

<i>Phương pháp đặc tả</i>	<i>Thể loại</i>	<i>Điểm mạnh</i>	<i>Điểm yếu</i>
- Ngôn ngữ tự nhiên	Không hình thức	<ul style="list-style-type: none">- Dễ học- Dễ sử dụng- Dễ hiểu đối với khách hàng.	<ul style="list-style-type: none">- Không chính xác- Không rõ ràng, mâu thuẫn và không đầy đủ.
<ul style="list-style-type: none">- Mô hình thực thể quan hệ- Phân tích hệ thống	Bán hình thức	<ul style="list-style-type: none">- Khách hàng có thể hiểu.- Chính xác hơn các phương pháp không hình thức	<ul style="list-style-type: none">- Không chính xác như các phương pháp hình thức.- Khó định lượng thời gian.
<ul style="list-style-type: none">- Máy hữu hạn trạng thái- Mạng Petri	Hình thức	<ul style="list-style-type: none">- Cực kỳ chính xác- Có thể giảm các lỗi đặc tả- Có thể giảm chi phí và nhân lực- Có thể hỗ trợ việc chứng minh tính chính xác	<ul style="list-style-type: none">- Khó học- Khó sử dụng- Khách hàng hầu như không hiểu được

