SOFTWARE DESIGN

Công nghệ phần mềm nâng cao **Nhóm: 2**

Thành viên:
Phan Văn Thanh
Lê Thị Len
Hoàng Thị Vân Anh

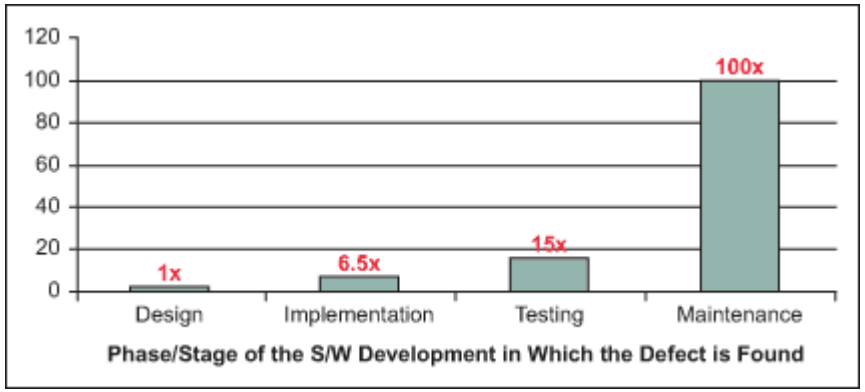
Giới thiệu

- Thiết kế vừa là quy trình vừa là kết quả của một quy trình.

✓ **Định nghĩa 1:** thiết kế là quy trình trình định nghĩa kiến trúc, thành phần, interfaces và các thuộc tính khác của một hệ thống hoặc một thành phần.

✓ **Định nghĩa 2:** thiết kế phần mềm là kết quả của một quy trình, kết quả này mô tả kiến trúc của một phần mềm như làm thế nào để phân rã và tổ chức trong các thành phần và các giao tiếp giữa các thành phần như thế nào.

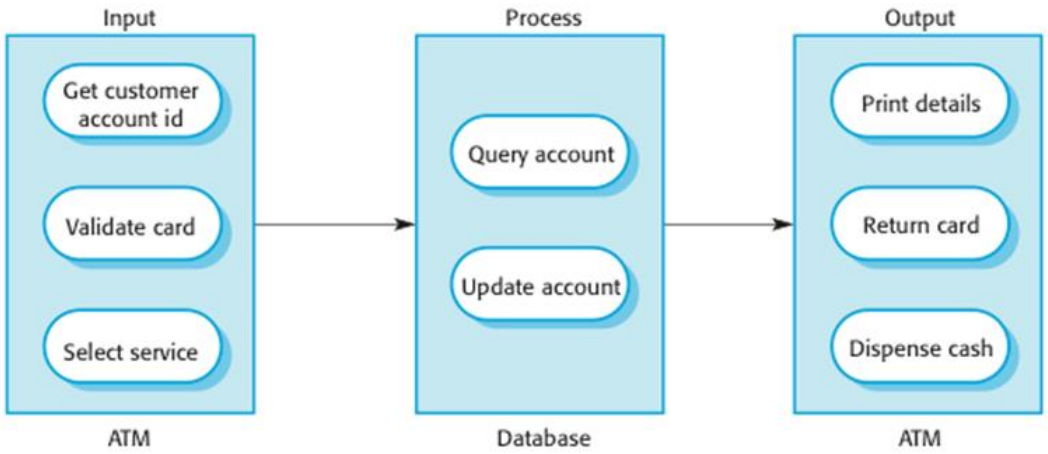
- Thiết kế phần mềm đóng vai trò quan trọng.



Hình 1. Chi phí khi phát hiện ra lỗi

Hoạt động trong thiết kế phần mềm

- Thiết kế phần mềm bao gồm 2 hoạt động:
- ✓ Thiết kế kiến trúc (còn được gọi là thiết kế mức cao): là phát triển mức kiến trúc cao nhất và tổ chức của phần mềm và chỉ ra các thành phần khác nhau trong hệ thống.
- ✓ Thiết kế chi tiết: chỉ ra chi tiết và đầy đủ mỗi thành phần để có thể xây dựng phần mềm.



Hình 1 Kiến trúc hệ thống ATM

1. Nguyên tắc thiết kế phần mềm cơ bản

- ❑ Thiết kế có thể được xem như một hình thức giải quyết vấn đề.
- ❑ Thiết kế phần mềm là một phần quan trọng của quy trình phát triển phần mềm
- ❑ Quy trình thiết kế phần mềm thường có 2 bước:
- ✓ Thiết kế kiến trúc: mô tả phần mềm tổ chức các phần mềm như thế nào.
- ✓ Thiết kế chi tiết mô tả hành động mong muốn của từng thành phần

Đầu ra của 2 quy trình này là tập mô hình và tài liệu ghi lại những quyết định cũng lời giải thích cho mỗi lý do quyết định.

1. Nguyên tắc thiết kế phần mềm cơ bản

- Nguyên tắc thết kế phần mềm bao gồm
- ✓ Trừu tượng hóa: nhìn vào đối tượng và các thông tin liên quan để cụ thể hóa mục đích và tránh bỏ xót thông tin.
- ✓ Ghép nối và liên kết: Ghép nối là một độ đo của độ phụ thuộc lẫn nhau giữa các module trong chương trình máy tính, trong khi đó liên kết là độ đo độ mạnh của mối liên kết giữa các phần tử trong một module.
- ✓ Phân rã hóa và module hóa: phần mềm lớn được chia thành một số thành phần định danh và mô tả các tương tác giữa các thành phần.
- ✓ Đói gói và ẩn thông tin: nhóm và đóng gói chi tiết bên trong của một trừu tượng và làm cho dữ liệu không thể được truy cập từ bên ngoài
- ✓ Tính đầy đủ, toàn vẹn và nguyên thủy
- ✓ Tách liên quan: một liên quan thiết kế là một lĩnh vực thiết kế mà liên quan đến một hay nhiều các bên liên quan. Mỗi kiến trúc nhìn một hay nhiều khung nhìn liên quan. Tách liên quan bởi những khung nhìn cho phép quan tâm các bên liên quan để tập trung vào một việc tại một thời điểm và yêu cầu, chung cấp phương tiện quản lý

2. Những vấn đề chính trong thiết kế kiến trúc

- Truy cập đồng thời: giải quyết các vấn đề về tính hiệu quả, tính nguyên tố, tính đồng bộ hóa và lập kế hoạch
- Điều khiển và xử lý các sự kiện: làm thế nào để tổ chức dữ liệu và dòng dữ liệu; làm thế nào xử lý các sự kiện tạm thời và các phản xạ qua các cơ chế lời gọi ngầm và gọi lại
- Dữ liệu bền vững: làm thế nào để xử lý dữ liệu tồn tại lâu dài
- Phân phối các thành phần: làm thế nào để phần phối các phần mềm trên phần cứng (bao gồm phần cứng máy tính và phần cứng mạng), làm thế nào các phần mềm giao tiếp được với nhau, và làm thế nào xử lý với không tương thích phần mềm.

2.Những vấn đề chính trong thiết kế kiến trúc

- Lỗi và xử lý ngoại lệ và chịu lỗi: làm thế nào để phòng chống, chịu đựng lỗi và các xử lý với các điều kiện ngoại lệ
- Tương tác và trình bày: làm thế nào để cấu trúc và tổ chức tương tác người dung và biểu diễn thông tin.
- Bảo mật: làm thế nào để ngăn chặn tiết lộ trái phép, thay đổi, xóa hoặc từ chối truy cập thông tin từ các nguồn khác; làm thế nào để hạn chế thiết hại của cuộc tấn công hoặc xâm phạm (tốc độ sữa chữa và phục hồi, phục hồi an toàn)

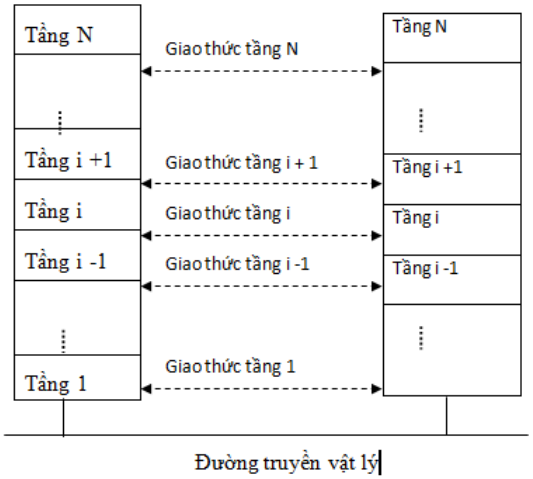
3. Kiến trúc và cấu trúc phần mềm

- Kiến trúc phần mềm là tập hợp các cấu trúc cần thiết để suy luận về hệ thống, trong đó có các yêu tố phần mềm, mối quan hệ giữa chúng và đặc tính của cả hai.
- Cấu trúc và góc nhìn: góc nhìn biểu diễn một phần khía cạnh của kiến trúc phần mềm. Mỗi góc nhìn cung cấp những vấn đề khác nhau liên quan đến phần mềm.
  - ✓ Góc nhìn logic: đáp ứng các yêu cầu chức năng
  - ✓ Góc nhìn tiến trình: vấn đề đồng thời
  - ✓ Góc nhìn vật lý: vấn đề phần phối
  - ✓ Góc nhìn phát triển: làm thế nào để chia nhỏ thành phần thành các thành phần đơn vị có thể thực hiện được

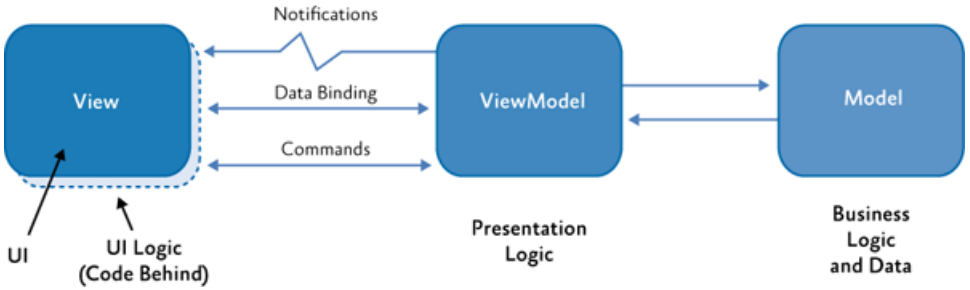
3. Kiến trúc và cấu trúc phần mềm

- Kiểu kiến trúc: có thể coi là một số mẫu kiến trúc. Một số kiến trúc chính như sau:
  - ✓ Kiến trúc thường (phân tầng, pipes and filter, blackboard)
  - ✓ Các hệ thống phân tán (client-server, three-tiers, broker)
  - ✓ Các hệ thống tương tác (MVC, MVVM)
  - ✓ Các hệ thống mô phỏng (microkernel, reflection)
  - ✓ Các kiểu khác (batch, interperters, process control, rule-based)

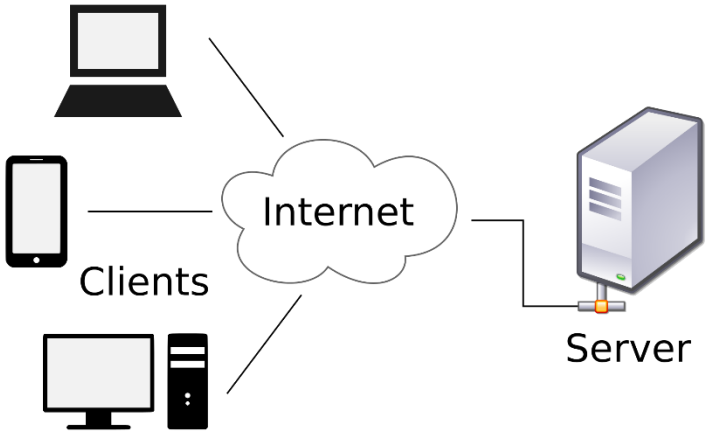
3.1: Kiểu kiến trúc



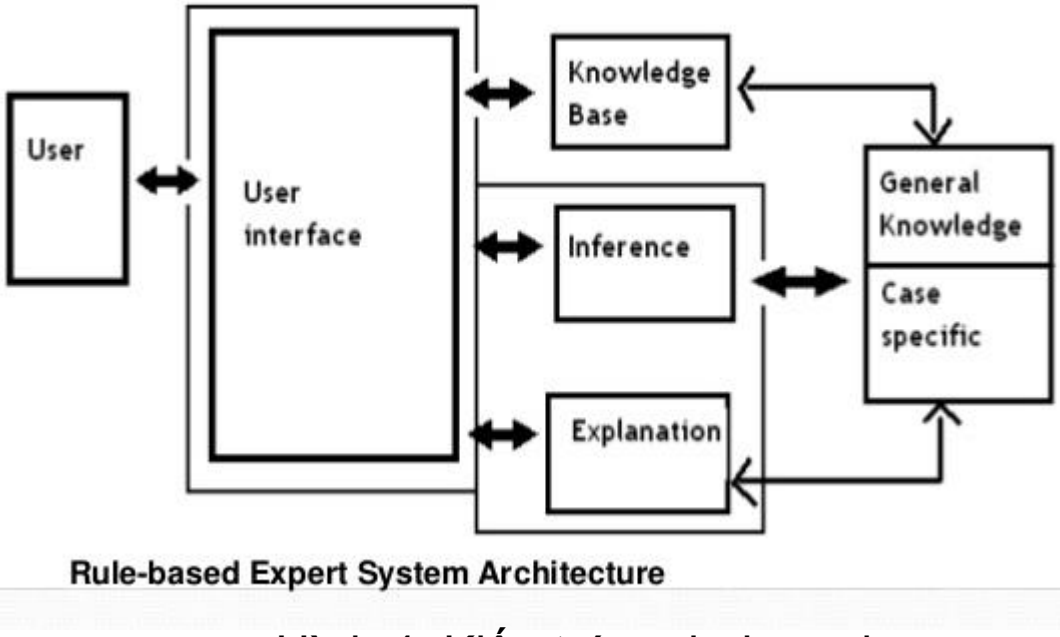
Hình 1: Kiến trúc phần tầng



Hình 3: Model – View – View Model



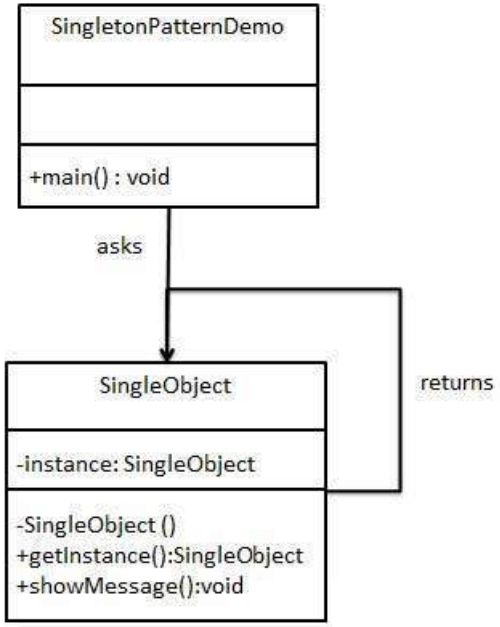
Hình 2: Kiến trúc client-server



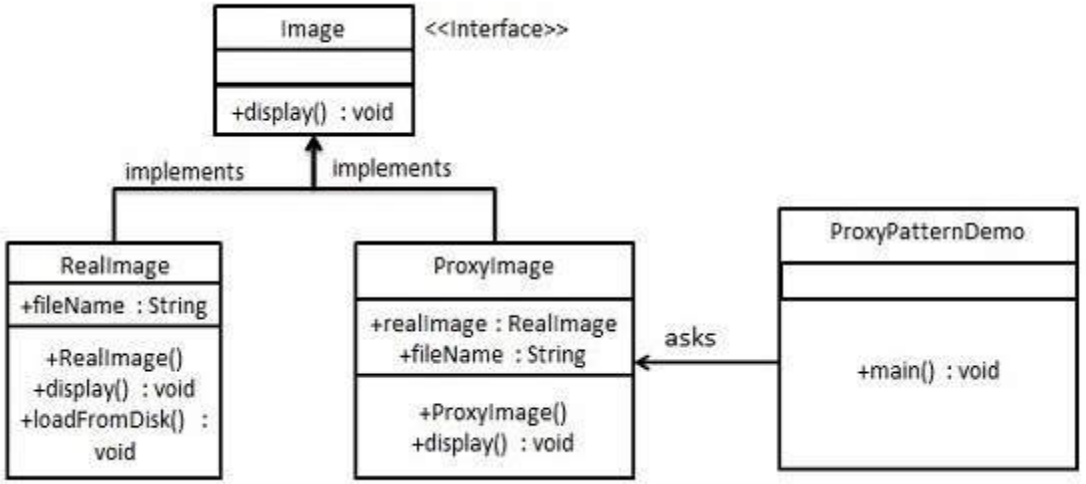
Hình 4: Kiến trúc rule-based

3. Kiến trúc và cấu trúc phần mềm

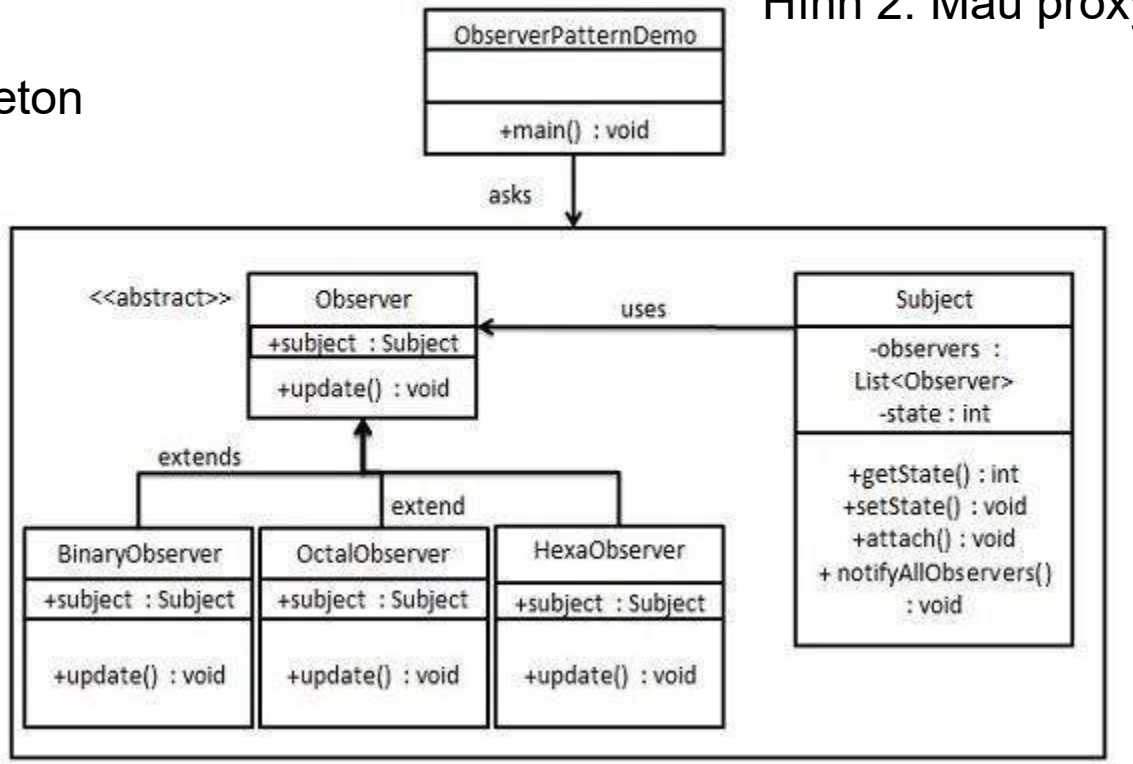
- Mẫu thiết kế: là một số giải pháp phổ biến để giải quyết các vấn đề phổ biến được đưa ra. Mẫu thiết kế có thể sử dụng mô tả thiết kế cụ thể. Các mẫu thiết kế bao gồm:
- ✓ Mẫu tạo (builder, factory, prototype, singleton)
- ✓ Mẫu cấu trúc (adapter, bridge, composite, decorator, façade, fly-weith, proxy)
- ✓ Mẫu hành vi (command, interpreter, interaction, observer, stragegy, state, template, visitor)



Hình 1: Mẫu singleton



Hình 2: Mẫu proxy



3. Kiến trúc và cấu trúc phần mềm

- Tính tương tự giữa chương trình và framework.

→ Một cách để thiết kế phần mềm đó là sử dụng lại những thiết kế phần mềm và cách thành phần trong những chương trình tương tự. Framework sẽ là trợ thủ đắc lực cho việc phát triển phần mềm làm rút ngắn thời gian phát triển và tăng khả năng bảo trì.

4. Thiết kế giao diện người dùng

Tại sao lại cần phải làm thiết kế giao diện?

- Thiết kế giao diện người dùng là một phần quan trọng quá trình thiết kế phần mềm.
- Thiết kế giao diện và xử lý tương tác với người sử dụng là một yếu tố quan trọng trong việc sử dụng phần mềm.
- Người thiết kế phải làm sao để phù hợp với kĩ năng, kinh nghiệm và mong đợi từ phía người sử dụng phần mềm.

4. Thiết kế giao diện người dùng

Nội dung

- **Nguyên tắc cơ bản trong thiết kế**
- **Các vấn đề khi thiết kế**
- **Các kiểu tương tác**
- **Cách biểu diễn thông tin**
- **Quy trình thiết kế**

4. Thiết kế giao diện người dùng

4.1: Nguyên tắc cơ bản trong thiết kế

- **Dễ học:** Phần mềm cần phải dễ học cách sử dụng, do đó người dùng có thể nhanh chóng bắt đầu làm việc sử dụng phần mềm đó
- **Quen thuộc với người sử dụng:** Giao diện nên dùng các thuật ngữ và khái niệm rút ra từ kinh nghiệm của những người sẽ dùng hệ thống nhiều nhất
- **Tính nhất quán:** giao diện cần nhất quán sao cho các thao tác gần giống nhau có thể được kích hoạt theo cùng kiểu.
- **Ngạc nhiên tối thiểu:** Người dùng không bao giờ bị bất ngờ về hành vi của hệ thống

4. Thiết kế giao diện người dùng

4.1: Nguyên tắc cơ bản trong thiết kế

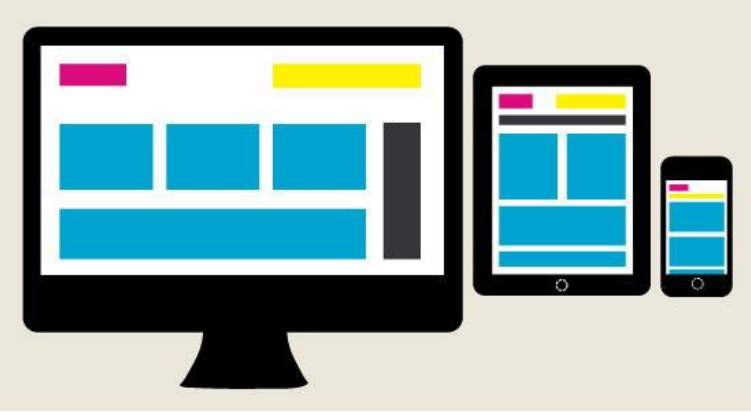
- **Khôi phục được:** Giao diện nên có các cơ chế cho phép người dùng khôi phục lại tình trạng hoạt động bình thường sau khi gặp lỗi
- **Hướng dẫn người dùng:** Giao diện nên có phản hồi có nghĩa khi xảy ra lỗi và cung cấp các tiện ích trợ giúp theo ngữ cảnh
- **Người dùng đa dạng:** Giao diện nên cung cấp các tiện ích tương tác thích hợp cho các loại người dùng hệ thống khác nhau

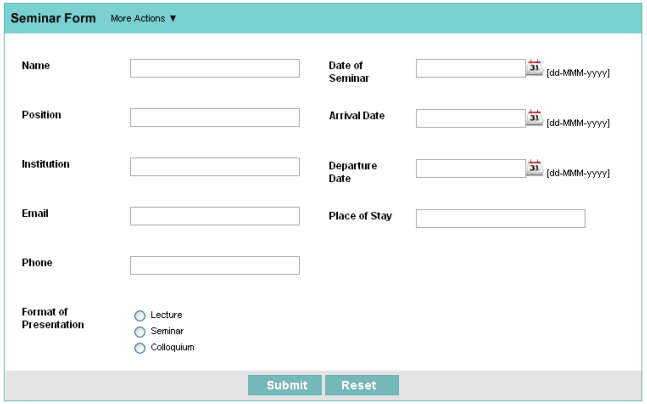
4. Thiết kế giao diện người dùng

4.2: Vấn đề trong thiết kế giao diện

Hai vấn đề cần xem xét:

- *Người dùng cung cấp thông tin cho hệ thống bằng cách nào?*
- *Hệ thống nên trình bày thông tin (output) cho người dùng như thế nào?*





4. Thiết kế giao diện người dùng

4.3: Các kiểu tương tác

- **Thao tác trực tiếp** – Direct manipulation
- **Chọn lựa bằng menu** – Menu selection
- **Điền form** – Form fill-in
- **Dòng lệnh** – Command language
- **Ngôn ngữ tự nhiên** – Natural language

Kiểu tương tác	Ưu điểm chính	Nhược điểm chính	Ví dụ
Thao tác trực tiếp	Tương tác trực quan, nhanh chóng và dễ hiểu	Có thể khó cài đặt. Chỉ thích hợp khi có ẩn dụ hình ảnh cho các tác vụ và đối tượng	Trò chơi điện tử và các ứng dụng có drag & drop
Chọn lựa bằng menu	Tránh lỗi cho người dùng, không phải làm nhiều thao tác	Chậm chạp với người sử dụng có kinh nghiệm. Có thể phức tạp nếu có nhiều lựa chọn menu	Đa số các hệ thống thông dụng
Điền form	Nhập dữ liệu đơn giản, dễ học, có thể kiểm tra được	Tốn không gian hiển thị, rắc rối khi lựa chọn của người dùng không khớp với kiểu dữ liệu của form	Đăng kí thông tin cá nhân, khai thuế
Dòng lệnh	Mạnh và linh động	Khó học, xử lý lỗi kém	Terminal, Autocad
Ngôn ngữ tự nhiên	Đáp ứng được người dùng không chuyên, dễ mở rộng	Cần gõ nhiều, các hệ thống hiểu ngôn ngữ tự nhiên không đáng tin cậy	Trợ lý ảo

4. Thiết kế giao diện người dùng

4.4: Biểu diễn thông tin

Có 2 loại thông tin cần được biểu diễn:

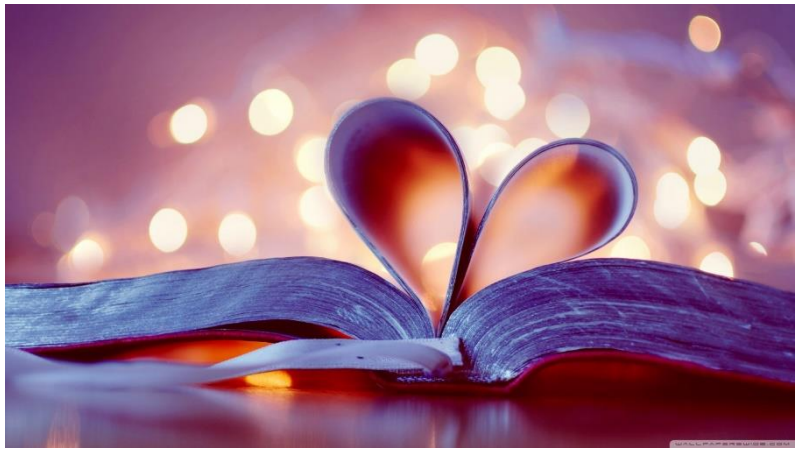
- *Thông tin tĩnh: Tạo ở lúc bắt đầu và không thay đổi trong phiên làm việc.*
- *Thông tin động: Thay đổi trong phiên làm việc và phải thông báo cho người sử dụng*

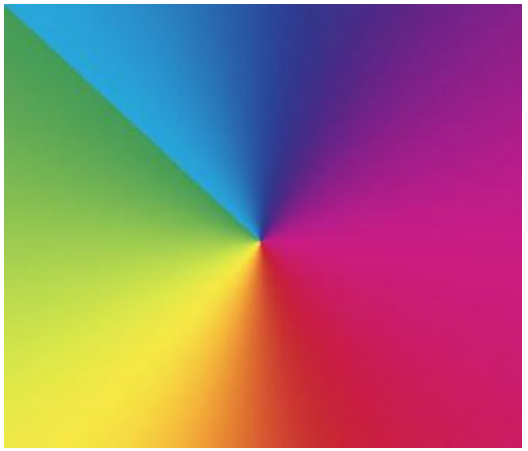
4. Thiết kế giao diện người dùng

4.4: Biểu diễn thông tin

Các kĩ thuật hiển thị lượng lớn thông tin:

- *Hình ảnh: có thể cho thấy quan hệ giữa các thực thể và các xu hướng của dữ liệu*
- *Màu sắc: thường dùng để highlight các thông tin đặc biệt*

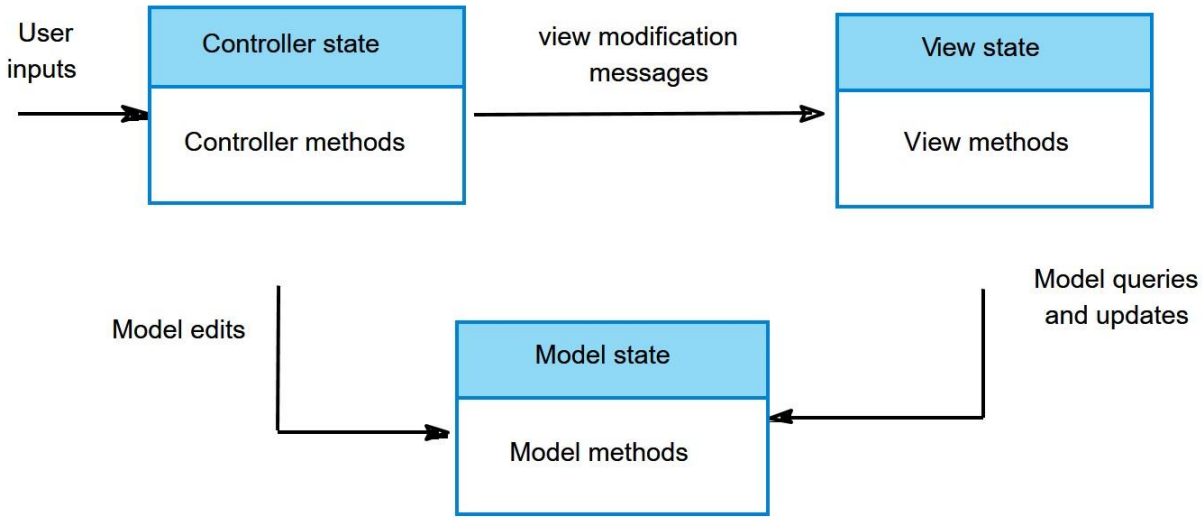




4. Thiết kế giao diện người dùng

4.4: Biểu diễn thông tin

Mô hình MVC hỗ trợ nhiều kiểu biểu diễn dữ liệu



4. Thiết kế giao diện người dùng

4.4: Biểu diễn thông tin

Chú ý khi sử dụng màu sắc

- *Hạn chế số màu và mức độ sặc sỡ*
- *Dùng sự thay đổi màu để báo hiệu thay đổi trạng thái hệ thống.*
- *Dùng kí hiệu màu (color coding) để hỗ trợ công việc người dùng đang cố làm. Highlight những điểm người dùng cần chú ý.*
- *Dùng kí hiệu màu một cách cẩn trọng và nhất quán*
- *Cẩn thận về hiệu ứng cặp đôi của màu sắc. Một số tổ hợp màu gây khó đọc, ví dụ như người ta không thể cùng lúc chú ý cả hai màu đỏ và xanh lam*

4. Thiết kế giao diện người dùng

4.5: Quy trình thiết kế giao diện

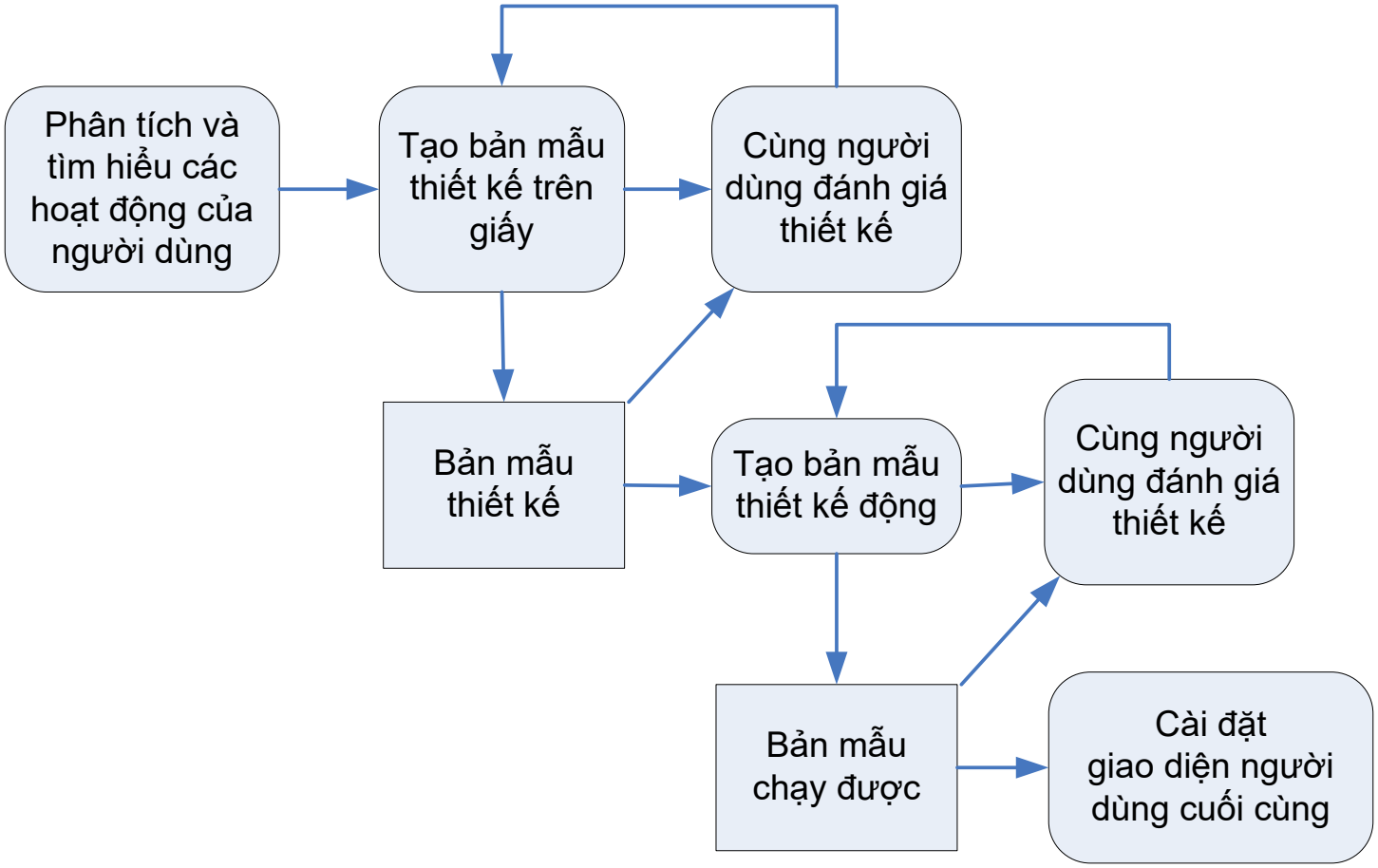
Thiết kế giao diện là một quy trình lặp đi lặp lại với sự liên lạc chặt chẽ giữa người dùng và người thiết kế.

Ba hoạt động chính trong quy trình:

- **User analysis**: *Tìm hiểu người dùng sẽ làm gì với hệ thống*
- **System prototyping**: *phát triển một loạt các bản mẫu để thử nghiệm*
- **Interface evaluation:** *thử nghiệm các bản mẫu cùng với người dùng*

4. Thiết kế giao diện người dùng

4.5: Quy trình thiết kế giao diện



5. Đánh giá chất lượng thiết kế

5.1: Đánh giá dựa trên thuộc tính sử dụng

Thuộc tính	Mô tả
Khả năng học	Người dùng mới cần bao lâu để có thể hoạt động hiệu quả với hệ thống?
Tốc độ vận hành	Tốc độ phản ứng của hệ thống có đáp ứng tốt công việc của người dùng?
Chịu lỗi	Mức độ dung thứ lỗi của hệ thống đối với lỗi người dùng.
Khả năng khôi phục	Khả năng hệ thống khôi phục từ lỗi của người dùng.
Tương thích	Hệ thống gắn bó chặt chẽ với một kiểu làm việc đến đâu?

5. Đánh giá chất lượng thiết kế

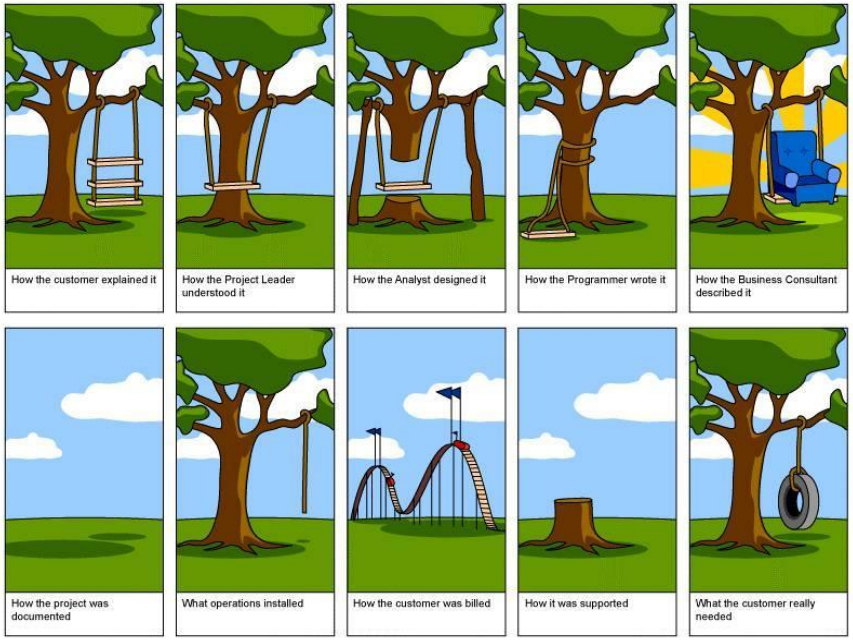
5.2: Kĩ thuật đánh giá và phân tích

- Câu hỏi điều tra để lấy phản hồi của người dùng.
- Quay video về việc sử dụng hệ thống rồi sau đó đánh giá nội dung.
- Cài các đoạn mã thu thập thông tin về các tiện ích được sử dụng và lỗi của người dùng.
- Phần mềm có chức năng thu thập phản hồi trực tuyến của người dùng

6. Mô hình thiết kế

Tại sao cần có các quy ước, ký hiệu, mô hình chung trong thiết kế phần mềm?

- Truyền tải được nhiều thông tin.
- Tổ chức, trình bày trực quan và tạo nên các hệ thống phức tạp.
- Tất cả mọi người cùng hiểu được phần mềm xây dựng và hoạt động ra sao.



6. Mô hình thiết kế

Các loại ký hiệu, mô hình dùng trong thiết kế

- **Static view** - dùng trong thiết kế mức cao (kiến trúc, tổ chức)
- **Dynamic view** - dùng trong thiết kế chi tiết (hành vi, tương tác,..)

6. Mô hình thiết kế

6.1. Static view

- Architecture description languages (ADLs) – Ngôn ngữ đặc tả kiến trúc
- Class and object diagrams - Biểu đồ lớp
- Component diagrams - Biểu đồ thành phần
- Class responsibility collaborator cards (CRCs)
- Deployment diagrams - Biểu đồ triển khai
- Entity-relationship diagrams (ERDs) - Sơ đồ thực thể quan hệ
- Interface description languages (IDLs) - Ngôn ngữ mô tả giao diện
- Structure charts - Biểu đồ cấu trúc

6. Mô hình thiết kế

6.2. Dynamic view

- Activity diagram – Sơ đồ hoạt động
- ommunication diagram - Biểu đồ giao tiếp
- Data flow diagrams (DFDs) - Sơ đồ luồng dữ liệu
- Decision tables and diagrams - Bảng và biểu đồ quyết định
- Flowcharts - Biểu đồ tiến trình (lưu đồ)
- Sequence diagrams - Biểu đồ tuần tự
- State transition and state chart diagrams - Biểu đồ trạng thái
- Formal specification languages - Ngôn ngữ đặc tả chính thức
- Pseudo code and program design languages (PDLs) - Mã giả và ngôn ngữ thiết kế chương trình

7. Chiến lược và phương pháp

7.1. Thiết kế hướng chức năng

- Bản thiết kế được phân giải thành một **bộ các đơn thể** được **tác động lẫn nhau**, mỗi đơn thể có một **chức năng được xác định rõ ràng**.
- Dùng biểu đồ dòng dữ liệu, lược đồ cấu trúc
- Nhược điểm:
  - Nếu thay đổi một chức năng có thể làm ảnh hưởng đến những chức năng khác
- Ưu điểm:
  - Phù hợp cho hệ thống có khối lượng thông tin trạng thái nhỏ nhất và thông tin dùng chung giữa các đơn thể là rõ ràng nhất

7. Chiến lược và phương pháp

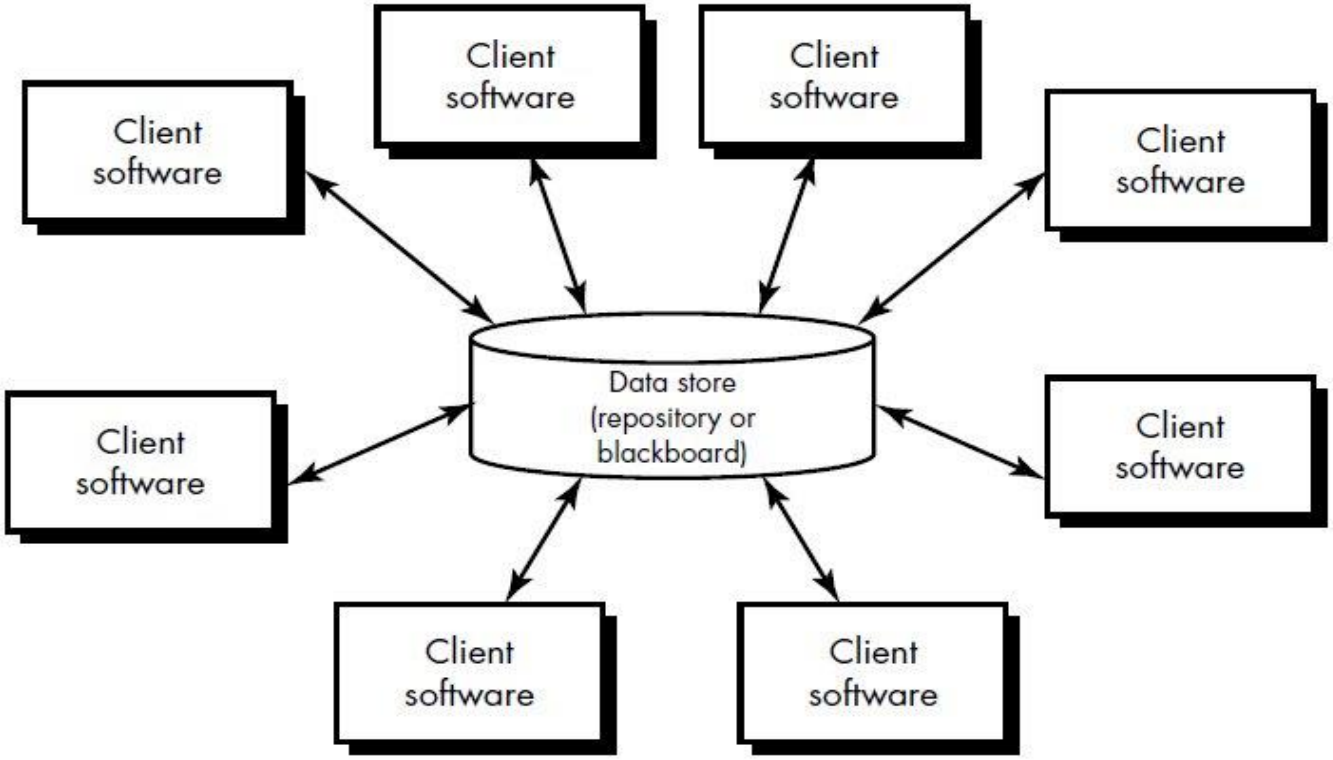
7.2. Thiết kế hướng đối tượng

- Nhìn nhận hệ thống như một **bộ các đối tượng**. Mỗi đối tượng có những thông tin trạng thái riêng của nó.
- Dựa trên việc che dấu dữ liệu
- Các đố lượng độc lập, sẵn sàng thay đổi
- Ưu điểm
  - Dễ bảo trì
  - Có thể dùng lại
  - Thiết kế dễ hiểu
- Nhược điểm
  - Cách nhìn hệ thống tự nhiên là hướng chức năng nên khó phân chia ra các đối tượng thích hợp

7. Chiến lược và phương pháp

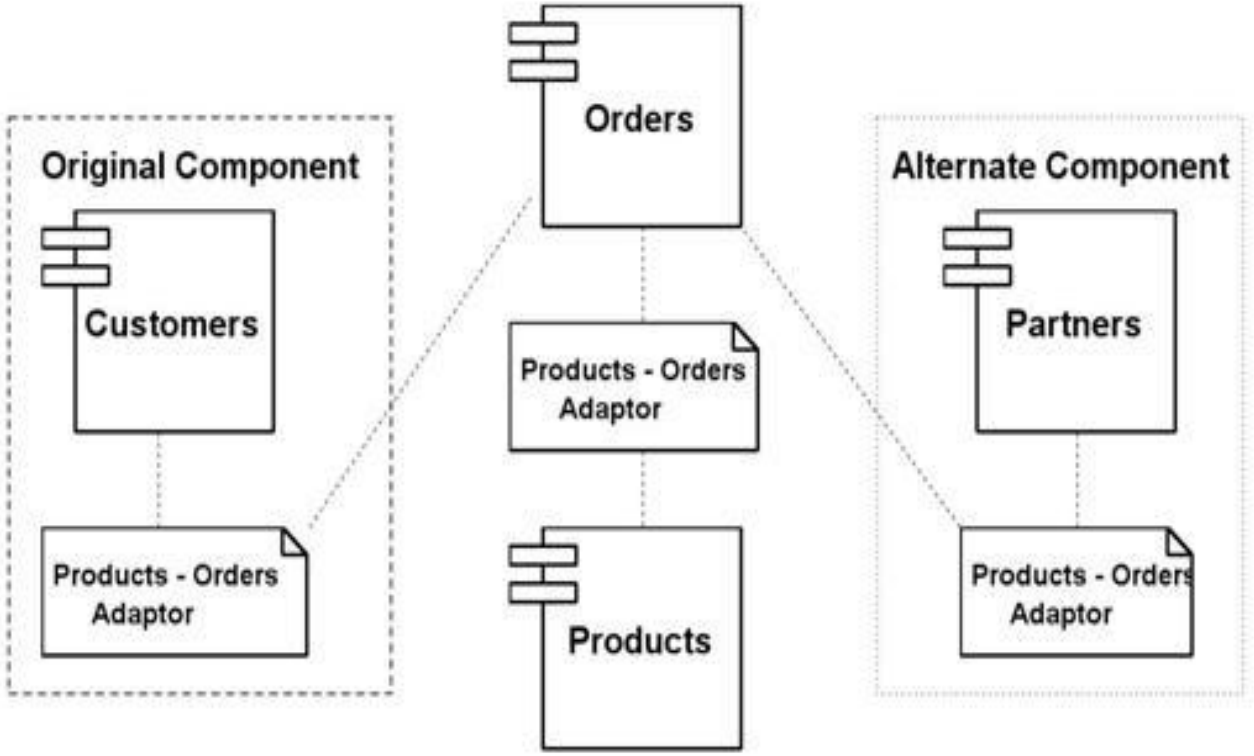
7.3. Thiết kế lấy cấu trúc dữ liệu làm trung tâm

- Mô tả đầu vào và đầu ra của cấu trúc dữ liệu → phát triển cấu trúc điều khiển của phần mềm



7. Chiến lược và phương pháp

7.4. Thiết kế hướng thành phần

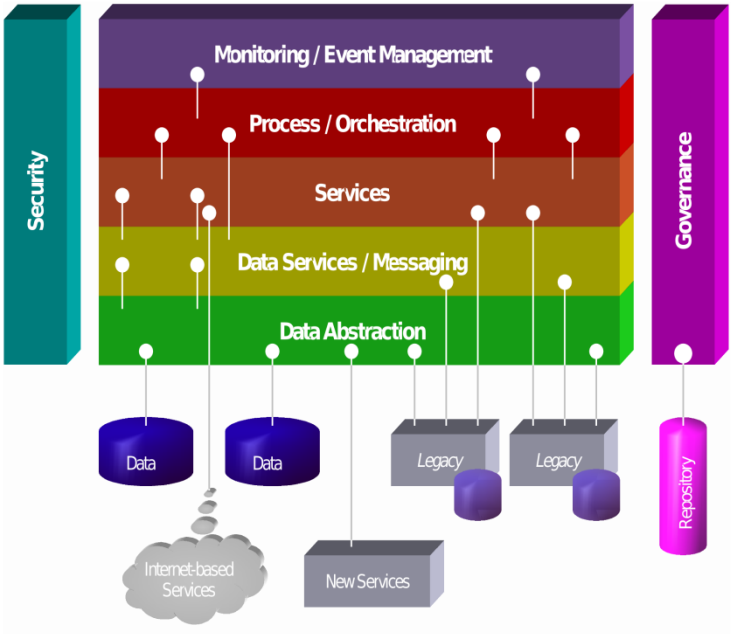


7. Chiến lược và phương pháp

7.5. Các phương pháp khác

- Service-Oriented Architecture - Kiến trúc hướng dịch vụ tập hợp các dịch vụ kết nối mềm dẻo với nhau.

Tái sử dụng phần mềm, linh hoạt khi mở rộng, kết nối và tích hợp.



8. Công cụ thiết kế phần mềm

- Chuyển các yêu cầu phần mềm thành một mô hình thiết kế trực quan
- Cung cấp mô tả tổng quan đến chi tiết của phần mềm
- Trợ giúp cho quá trình đánh giá chất lượng phần mềm

Thank You !