Giai đoạn 2
Phân tích hệ thống

Chương 6
Chuyển sang thiết kế hệ thống

Nội dung

- Đánh giá các giải pháp phần mềm và các chiến lược phát triển.
- Ưu điểm và nhược điểm giữa việc tự phát triển phần mềm và việc mua/chỉnh sửa một gói phần mềm.
- Tài nguyên bên ngoài và các ứng dụng cho người sử dụng.
- Các bước mua và đánh giá một gói phần mềm.
- *RFP* và *RFQ*.
- Tài liệu các yêu cầu hệ thống và trình bày cho ban quản lý.

Nội dung

- Chuyển từ phân tích sang thiết kế.
- Sự khác biệt giữa thiết kế luận lý và thiết kế vật lý.
- Tầm quan trọng của việc làm bản mẫu; các phương pháp làm bản mẫu, các công cụ và kỹ thuật.
- Quá trình thiết kế hệ thống và các hướng dẫn thiết kế hệ thống.
- Tạo và sử dụng các mã thích hợp trong quá trình thiết kế và phát triển hệ thống.

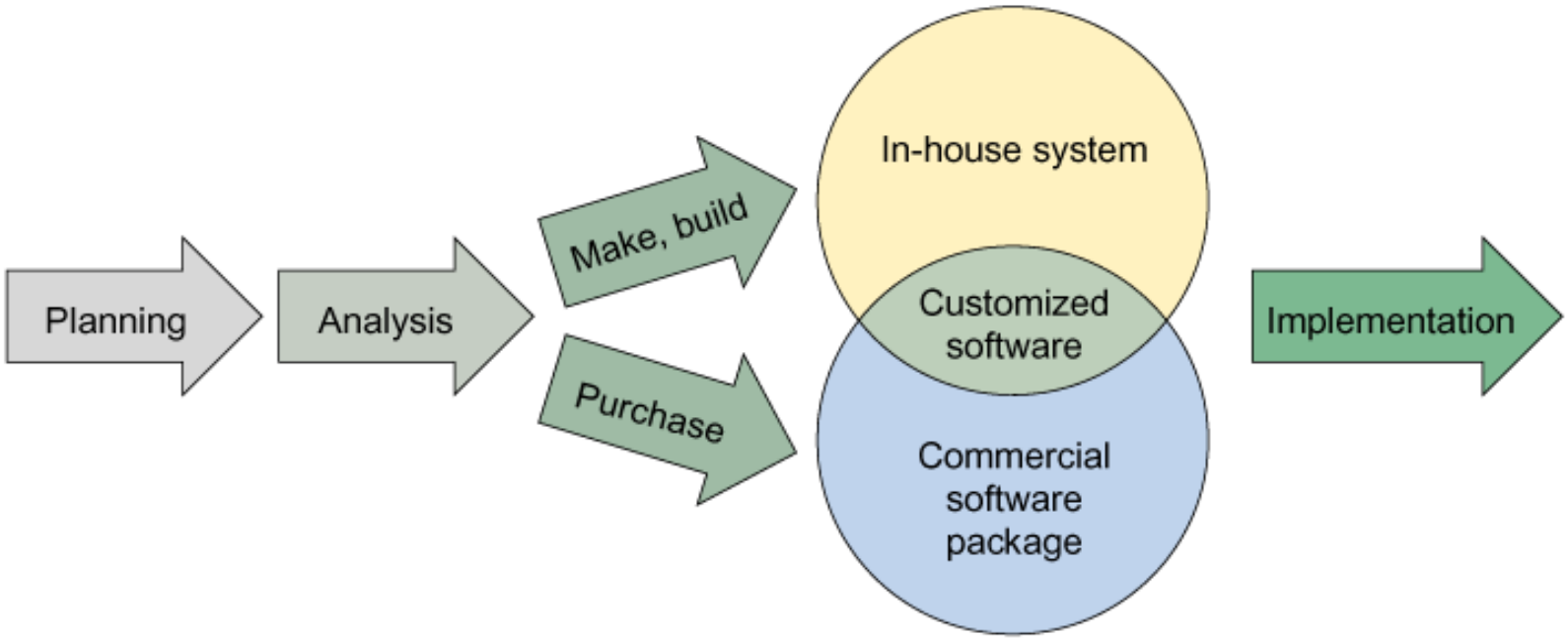
Giới thiệu

- Chương này trình bày các công việc còn lại trong giai đoạn phân tích hệ thống
  - Đánh giá các giải pháp khác nhau.
  - Chuẩn bị tài liệu yêu cầu hệ thống.
  - Trình bày cho ban quản lý.

Đánh giá các giải pháp phần mềm

- **Quyết định tự phát triển hoặc mua phần mềm**
  - **Phần mềm tự phát triển**
    - Được phát triển bởi phòng CNTT của công ty.
  - **Gói phần mềm**
    - Được mua hoặc thuê từ các hãng hoặc nhà cung cấp phần mềm.
    - Ứng dụng ngang.
    - Ứng dụng dọc.

Đánh giá các giải pháp phần mềm



Hình 6.1. Dựa vào giải pháp nào được chọn, công ty có thể tự phát triển hệ thống hoặc mua và có thể chỉnh sửa gói phần mềm thương mại.

Đánh giá các giải pháp phần mềm

- **Tự phát triển phần mềm**
  - *developing software in-house*
  - **Các lý do tự phát triển phần mềm:**
    - Thỏa mãn các yêu cầu riêng biệt.
    - Giảm thiểu các thay đổi trong các chính sách và các thủ tục nghiệp vụ
    - Thỏa mãn các ràng buộc của hệ thống hiện tại.
    - Thỏa mãn các ràng buộc của công nghệ hiện tại.
    - Phát triển các tài nguyên và các năng lực bên trong.

Đánh giá các giải pháp phần mềm

- **Mua gói phần mềm**
  - *purchasing a software package*
  - Các lý do mua gói phần mềm:
    - Chi phí thấp hơn.
    - Thời gian có được nhanh hơn.
    - Độ tin cậy và các chuẩn hiệu suất đã được kiểm tra.
    - Ít nhân viên phát triển kỹ thuật.
    - Các nâng cấp trong tương lai được cung cấp từ nhà cung cấp.
    - Các công ty khác được xem là các tài nguyên.

Đánh giá các giải pháp phần mềm

- **Chỉnh sửa gói phần mềm**
  - ***customizing software package***
  - **Mua gói phần mềm cơ bản để có thể chỉnh sửa nhằm thỏa mãn các yêu cầu.**
  - **Thương lượng với nhà cung cấp phần mềm để thực hiện các phần mở rộng nhằm thỏa mãn các yêu cầu .**
  - **Mua gói phần mềm và tự chỉnh sửa.**

Đánh giá các giải pháp phần mềm

- **Nhà cung cấp dịch vụ ứng dụng**
  - Cung cấp các ứng dụng bằng cách trả phí sử dụng hoặc lệ phí.
  - Dịch vụ cung cấp được gọi là *application hosting*.
  - Cho thuê các ứng dụng.
- **Tài nguyên bên ngoài**
  - *outsourcing*
  - Sử dụng các công ty bên ngoài để xử lý một phần tải làm việc, ngắn hạn hoặc dài hạn.
  - Ký kết với các hãng nhân sự.
  - Các hãng quản lý hệ thống hoặc quản lý phương tiện.

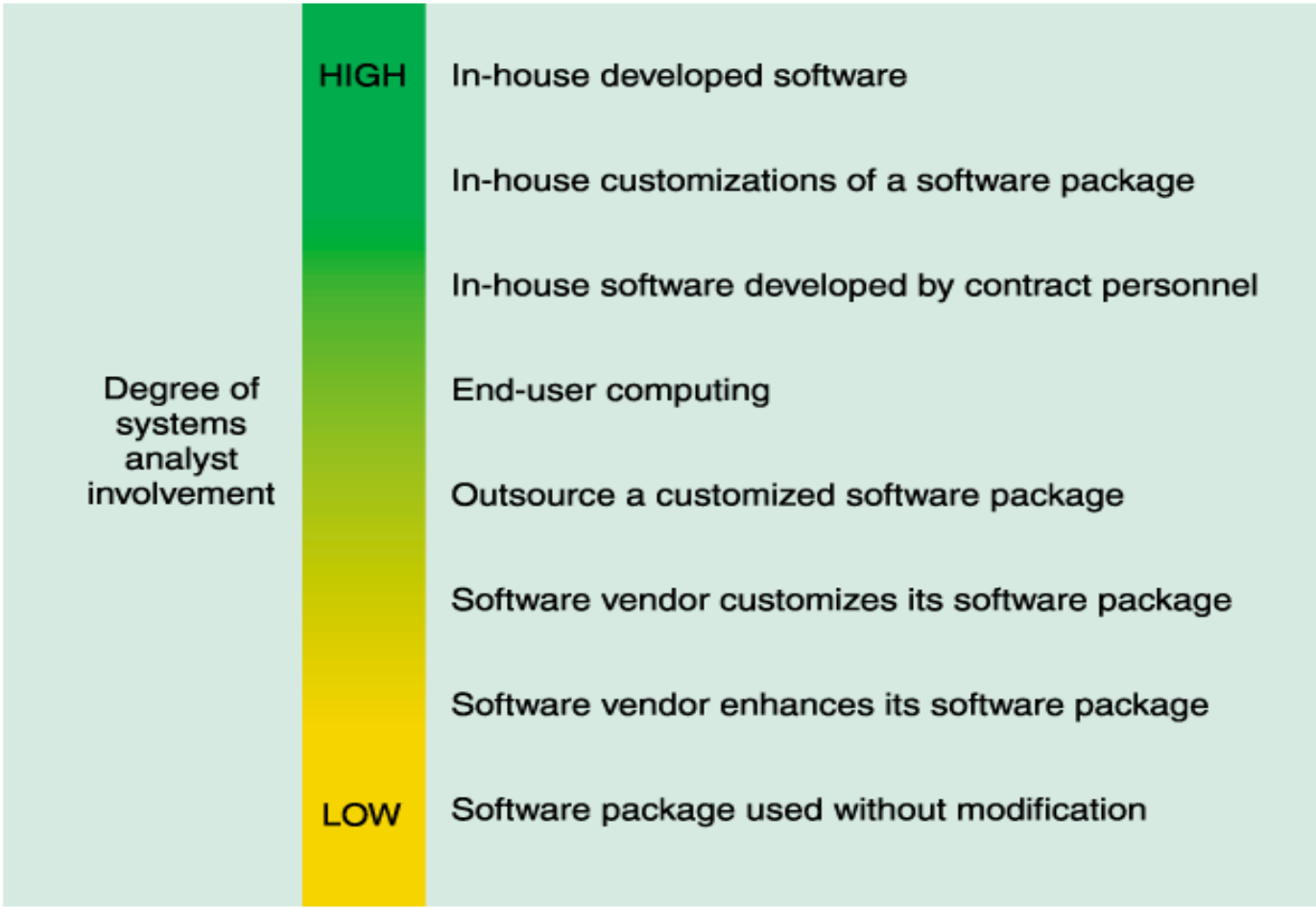
Đánh giá các giải pháp phần mềm

- **Hệ thống của người sử dụng cuối cùng**
  - *end-user system*
  - Sử dụng các phần mềm nghiệp vụ chuẩn.
  - Có thể cung cấp các giải pháp đơn giản, chi phí thấp.
  - Người sử dụng có thể thiết kế các biểu mẫu nhập dữ liệu và các báo cáo riêng.

Đánh giá các giải pháp phần mềm

- **Chọn giải pháp phần mềm**
  - Quyết định sẽ ảnh hưởng đến các giai đoạn còn lại của *SDLC*.
  - Công việc của người phân tích hệ thống phụ thuộc vào giải pháp được chọn.

Đánh giá các giải pháp phần mềm



Hình 6.2. Tầm quan trọng của người phân tích phụ thuộc vào giải pháp nào được chọn.

Các bước đánh giá và mua gói phần mềm

- **Quá trình 5 bước**
  1. Đánh giá các yêu cầu HTTT.
  2. Xác định nhà cung cấp phần mềm có năng lực.
  3. Đánh giá các giải pháp phần mềm.
  4. Mua gói phần mềm.
  5. Cài đặt gói phần mềm.

Các bước đánh giá và mua gói phần mềm

- **Bước 1: đánh giá các yêu cầu HTTT.**
  - Xác định các tính năng chính của hệ thống.
  - Đánh giá khối lượng và sự gia tăng trong tương lai.
  - Mô tả các ràng buộc phần cứng.
  - Chuẩn bị yêu cầu đề nghị (RFP – *Request For Proposal*).

Các bước đánh giá và mua gói phần mềm

- **Bước 2: xác định nhà cung cấp phần mềm có năng lực.**
  - Bước kế tiếp là liên hệ với nhà cung cấp có năng lực.
  - *RFP* sẽ giúp nhà cung cấp xác định các giải pháp.
  - Các nguồn thông tin khác nhau về nhà cung cấp:
    - Người bán lẻ
    - Nhà sản xuất máy tính
    - Báo thương mại công nghiệp hoặc các *Web site*
    - Nhóm tư vấn CNTT

Các bước đánh giá và mua gói phần mềm

- **Bước 3: đánh giá các giải pháp phần mềm.**
  - Mục đích là so sánh các gói phần mềm và chọn gói phần mềm tốt nhất.
  - Có được thông tin từ nhiều nguồn.
  - Quá trình đánh giá
    - Có được thông tin từ những người sử dụng hiện tại.
    - Chạy kiểm tra ứng dụng .
    - Định chuẩn gói phần mềm, nếu cần.

Các bước đánh giá và mua gói phần mềm

- **Bước 4: mua gói phần mềm.**
  - Bản quyền phần mềm (*software license*).
  - Hợp đồng thuê (*lease agreement*).
  - Hợp đồng bảo trì (*maintenance agreement*).

Các bước đánh giá và mua gói phần mềm

- **Bước 5: cài đặt gói phần mềm.**
  - Thời gian cài đặt phụ thuộc vào kích cỡ và độ phức tạp.
  - Trước khi sử dụng gói phần mềm, hoàn thành tất cả các bước thực hiện.
    - Nạp, cấu hình và kiểm tra phần mềm.
    - Đào tạo những người sử dụng.
    - Chuyển các tập tin dữ liệu thành các dạng thức mới.

Các bước đánh giá và mua gói phần mềm

- **Nhóm đánh giá và chọn lựa**
  - Mục đích của quá trình này là có được sản phẩm với chi phí sở hữu thấp nhất.
  - Cách tiếp cận nhóm bảo đảm các yếu tố quan trọng đều được xem xét và đưa ra sự chọn lựa đúng đắn.
  - Mục tiêu chính
    - Đánh giá các giải pháp hệ thống không chấp nhận.
    - Xếp hạng các giải pháp chấp nhận.
    - Trình bày các giải pháp cho ban quản lý để ra quyết định cuối cùng.

Hoàn tất phân tích hệ thống

- **Tài liệu các yêu cầu hệ thống**
  - Còn được gọi là mô tả các yêu cầu phần mềm.
  - Mô tả các giải pháp và đề nghị với ban quản lý
  - Tương tự như hợp đồng về điều sẽ được cung cấp.
  - Phải rõ ràng và dễ hiểu đối với người sử dụng.

Hoàn tất phân tích hệ thống

- **Trình bày cho ban quản lý**
  - **Năm quyết định có thể có:**
    1. **Tự phát triển hệ thống.**
    2. **Chỉnh sửa hệ thống hiện tại.**
    3. **Mua hoặc chỉnh sửa gói phần mềm.**
    4. **Thực hiện thêm công việc phân tích hệ thống.**
    5. **Chấm dứt toàn bộ công việc.**

Chuyển sang thiết kế hệ thống

- Điều chủ yếu là có tài liệu các yêu cầu hệ thống dễ hiểu và chính xác.
- Các lỗi sai, các thiếu sót và các điều mơ hồ sẽ ảnh hưởng đến chất lượng của sản phẩm cuối cùng.

Chuyển sang thiết kế hệ thống

- **Tổng quan về thiết kế hệ thống**
  - Thiết kế luận lý xác định các tính năng và các chức năng của hệ thống.
    - Còn được gọi là mô hình chính.
  - Thiết kế vật lý là một sơ đồ để thực hiện hệ thống.
- **Mối liên hệ giữa phân tích và thiết kế**
  - Giai đoạn thiết kế không thể bắt đầu cho đến khi kết thúc phân tích.
  - Chỉ nên trở về giai đoạn phân tích trong các tình huống hạn chế.

Làm bản mẫu

Làm bản mẫu

- *prototyping*
- Làm bản mẫu là một kỹ thuật thu thập thông tin.
- Làm bản mẫu hệ thống tạo ra một mô hình làm việc với đầy đủ tính năng của HTTT.
- Các bản mẫu có ích cho việc ghi nhận các phản ứng của người sử dụng, các đề nghị, các đổi mới và các kế hoạch sửa đổi.
- Bản mẫu là phiên bản của hệ thống được xây dựng nhanh.
- Bản mẫu là mô hình làm việc giúp người sử dụng hiểu hệ thống.

Làm bản mẫu

Làm bản mẫu

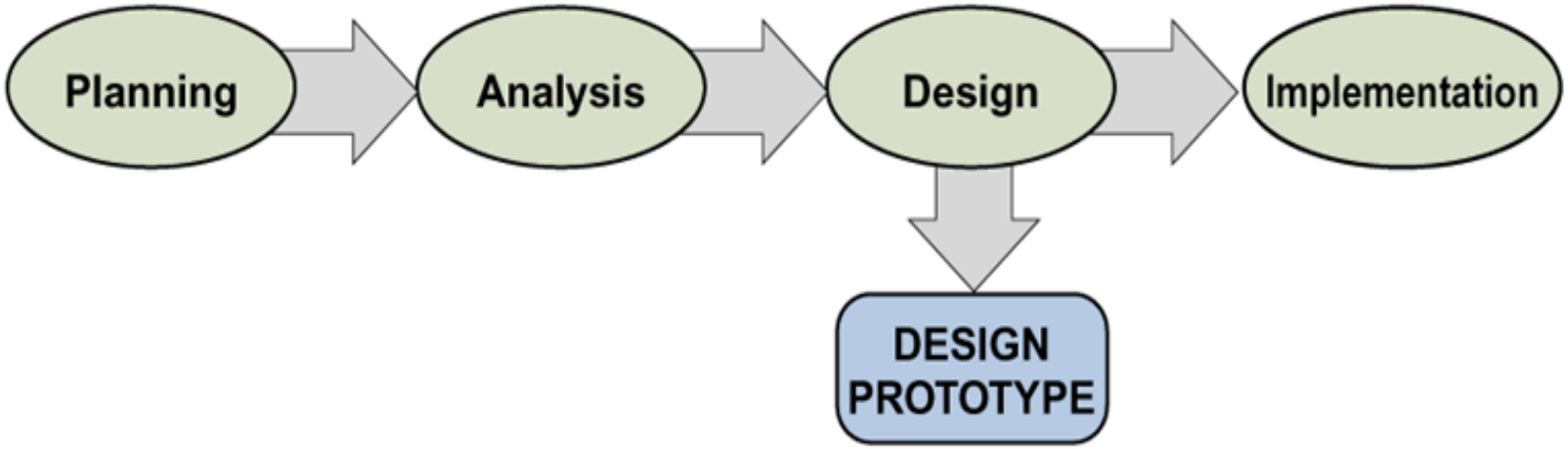
- Có thể loại bỏ các vấn đề trước khi có phiên bản cuối cùng.
- Người sử dụng có thể kiểm tra mô hình và chấp nhận hoặc yêu cầu sửa đổi.
- Có thể phát triển thành phiên bản cuối cùng của HTTT hoặc có thể chỉ được sử dụng để xác nhận các yêu cầu của người sử dụng và bị loại bỏ sau này.

Cách tiếp cận làm bản mẫu

Thiết kế bản mẫu

- Nhằm tạo ra bản mẫu thiết kế được người sử dụng chấp nhận để lập tài liệu và định chuẩn các tính năng của hệ thống cuối cùng.
- Có thể đạt được dữ liệu nhập của người sử dụng và sự chấp thuận của họ trong khi vẫn tiếp tục phát triển hệ thống.

Cách tiếp cận làm bản mẫu



Hình 6.3. Sản phẩm cuối cùng của làm bản mẫu là bản mẫu thiết kế được người sử dụng chấp thuận, trong đó ghi nhận các đặc điểm của hệ thống cuối cùng.

Làm bản mẫu

Lợi ích của làm bản mẫu

- Người sử dụng và người phát triển có thể tránh các điều hiểu nhầm.
- Người phát triển có thể tạo các mô tả chính xác.
- Người quản lý có thể đánh giá mô hình làm việc.
- Người phân tích có thể sử dụng bản mẫu để thực hiện kiểm tra và quá trình đào tạo.
- Làm bản mẫu giảm rũi ro xảy ra cho hệ thống cuối cùng.

Làm bản mẫu

- **Các vấn đề tiềm tàng của làm bản mẫu**
  - Tốc độ phát triển nhanh có thể phát sinh các vấn đề về chất lượng.
  - Các yêu cầu hệ thống khác không thể được kiểm tra một cách thỏa đáng.
  - Các bản mẫu phức tạp sẽ không thích đáng và khó quản lý.
- **Hạn chế của bản mẫu**
  - Kém hiệu quả hơn so với hệ thống được phát triển hoàn toàn.
  - Tốc độ xử lý chậm hơn và thời gian đáp ứng lâu hơn.
  - Có thể thiếu các yêu cầu bảo mật.

Làm bản mẫu

Các công cụ làm bản mẫu

- Các công cụ *CASE*
- Bộ tạo ứng dụng (*application generator*)
- Bộ tạo báo cáo (*report generator*)
- Bộ tạo màn hình (*screen generator*)
- Ngôn ngữ thế hệ thứ 4 (4GL-*Fourth-Generation Language*)
- Các công cụ làm bản mẫu thuộc môi trường thế hệ thứ 4

Làm bản mẫu

Bộ tạo ứng dụng

- *application generator*
- Còn được gọi là bộ tạo mã lệnh.
- Phát triển các chương trình nhanh chóng bằng cách chuyển từ mô hình luận lý trực tiếp thành mã lệnh 4GL.
- Ngôn ngữ điều khiển sự kiện mô tả các tác vụ mà chương trình phải thực hiện khi xảy ra các sự kiện nào đó.
- Ngôn ngữ phi thủ tục không đòi hỏi người lập trình viết chuỗi lệnh.
- Ngôn ngữ thủ tục đòi hỏi người lập trình tạo mã lệnh cho mỗi bước xử lý.

Làm bản mẫu

- **Bộ tạo báo cáo**
  - *report generator*
  - Còn được gọi là bộ viết báo cáo.
  - Thiết kế nhanh các báo cáo định dạng.
  - Có thể tạo báo cáo mẫu để xem trước khi thiết kế cuối cùng.
- **Bộ tạo màn hình**
  - *screen generator*
  - Còn được gọi là bộ tạo biểu mẫu.
  - Công cụ phần mềm tương tác.
  - Giúp thiết kế giao diện người sử dụng, tạo biểu mẫu màn hình và xử lý các thủ tục nhập dữ liệu.

Làm bản mẫu

❖ Các loại bản mẫu

- **Bản mẫu tạm thời** (***patched-up prototype***).
  - Đây là mô hình làm việc có tất cả các tính năng nhưng không hiệu quả.
  - Người sử dụng có thể tương tác với hệ thống.
  - Việc lưu trữ và truy xuất dữ liệu có thể không hiệu quả.
  - Có thể chỉ chứa các tính năng cơ bản.
- **Bản mẫu không vận hành được** (***non-operational prototpye***).
  - Đây là mô hình không vận hành được, ngoại trừ một số tính năng nào đó được kiểm tra.
  - Làm bản mẫu cho các mẫu nhập và các kết xuất.

Làm bản mẫu

- Các loại bản mẫu
  - **Bản mẫu loạt đầu tiên (*first-of-a-series prototype*)**
    - **Tạo hệ thống thử nghiệm (*pilot system*).**
    - **Bản mẫu là một mô hình vận hành được.**
    - **Có ích khi lập kế hoạch cài đặt cùng một hệ thống thông tin cho nhiều nơi. Ví dụ cài đặt, kiểm tra và sửa đổi hệ thống cho một nơi, và sau đó cài đặt tiếp tục cho các nơi khác.**

Làm bản mẫu

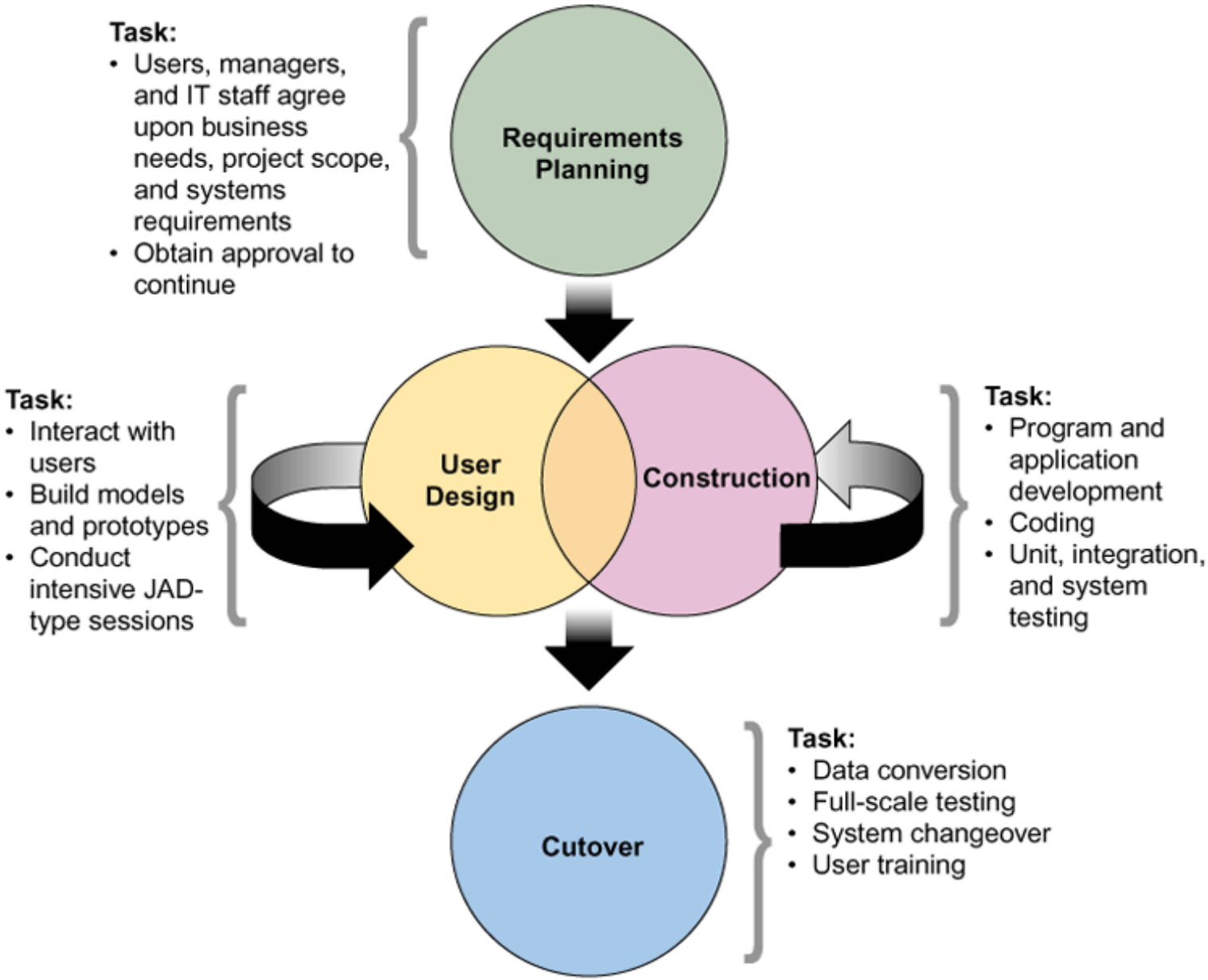
- Các loại bản mẫu
  - **Bản mẫu có các tính năng chính (*selected features prototype*)**
    - **Bản mẫu chỉ có một số tính năng chính của hệ thống. Sau khi các tính năng này được chấp nhận, bổ sung thêm các tính năng khác vào hệ thống.**
    - **Có thể chạy được một số chức năng trên trình đơn.**
    - **Hệ thống được xây dựng theo các đơn thể.**
    - **Đây là một phần của hệ thống thật.**

Làm bản mẫu

Hình 6.4. Sản phẩm cuối cùng của làm bản mẫu là mô hình làm việc của HTTT, sẵn sàng cho giai đoạn thực hiện.

Cách tiếp cận làm bản mẫu

- **Phát triển ứng dụng nhanh**
  - RAD - *Rapid Application Development*
  - RAD là một các tiếp cận hướng đối tượng để phát triển hệ thống, bao gồm phương pháp phát triển và các công cụ phần mềm.
  - RAD liên quan mật thiết với việc làm bản mẫu hệ thống.
  - Nhóm RAD xác định, phân tích, thiết kế và kiểm tra các bản mẫu.



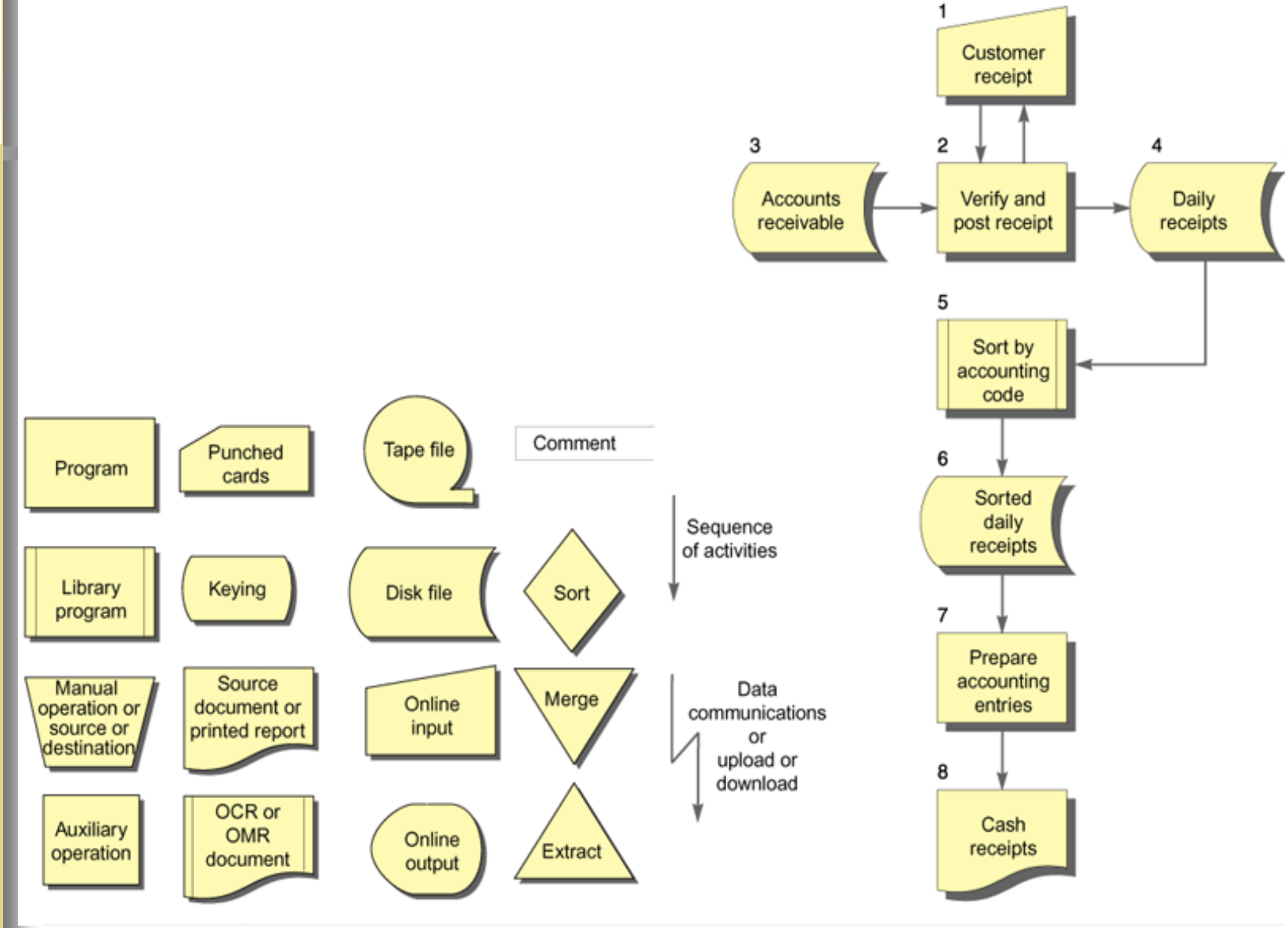
Hình 6.5. Mô hình RAD của James Martin.

Cách tiếp cận làm bản mẫu

- **RAD được sử dụng khi:**
  - Nhóm bao gồm những người phân tích và những người lập trình đã có kinh nghiệm về RAD.
  - Có các lý do cấp thiết để làm tăng tốc độ phát triển ứng dụng.
  - Dự án có ứng dụng thương mại điện tử mới lạ và cần có nhanh các kết quả.
  - Những người sử dụng thành thạo và gắn liền với các mục đích của công ty.

Các sơ đồ tiến trình của hệ thống

- **Công cụ mô hình hóa khác**
  - **Sơ đồ tiến trình của hệ thống (*system flowchart*)**
    - **Hiển thị quá trình chính và các tác vụ nhập, xuất.**
    - **Chủ yếu được sử dụng trong mô hình hóa vật lý.**
    - **Các ký hiệu khác nhau biểu diễn dữ liệu hoặc các tập tin trong môi trường vật lý cụ thể.**
    - **Hình dạng ký hiệu cho biết mục đích.**
    - **Các đường có mũi tên cho biết chiều đi của dữ liệu.**



Hình 6.6. Sử dụng các ký hiệu chuẩn ANSI, sơ đồ tiến trình của hệ thống cho thấy hệ thống đặt hàng xử lý việc trả tiền của khách hàng.

Tổng quan về thiết kế hệ thống

- Người phân tích phải hiểu toàn bộ thiết kế luận lý trước khi bắt đầu thiết kế vật lý
- Các bước thiết kế hệ thống.
  - Xem xét các yêu cầu hệ thống.
  - Thiết kế hệ thống
    - Xuất (*output*)
    - Nhập (*input*)
    - Các cơ sở dữ liệu và các tập tin
    - Kiến trúc hệ thống
- Trình bày thiết kế hệ thống.

Tổng quan về thiết kế hệ thống

- **Mục tiêu thiết kế hệ thống**
  - Xây dựng hệ thống sao cho:
    - Hiệu quả (*effective*)
    - Tin cậy (*reliable*)
    - Bảo trì được (*maintainable*)

Tổng quan về thiết kế hệ thống

- **Nghiên cứu thiết kế hệ thống**
  - **Nghiên cứu người sử dụng**
    - Làm cho hệ thống thân thiện với người sử dụng.
    - Lưu ý cách người sử dụng cung cấp dữ liệu cho hệ thống và nhận kết quả xuất từ hệ thống.
    - Đoán trước các yêu cầu trong tương lai.
      - Người sử dụng
      - HTTT
      - Công ty
    - Phải có tính linh động.

Tổng quan về thiết kế hệ thống

Nghiên cứu thiết kế hệ thống

Nghiên cứu dữ liệu

- Nhập dữ liệu tại nơi và vào lúc nó xảy ra.
- Kiểm tra dữ liệu khi nhập vào.
- Sử dụng các phương pháp nhập dữ liệu tự động
- Kiểm tra việc truy xuất khi nhập dữ liệu.
- Ghi nhận việc nhập hoặc thay đổi dữ liệu đối với các giá trị quan trọng.
- Nhập dữ liệu vào hệ thống chỉ một lần.
- Tránh trùng lặp dữ liệu.

Nghiên cứu việc xử lý

- Sử dụng thiết kế đơn thể.
- Thiết kế các đơn thể để thực hiện chức năng đơn lẻ.

Tổng quan về thiết kế hệ thống

- **Cân nhắc trong thiết kế**
  - Các mục đích thiết kế thường mâu thuẫn với nhau.
    - Việc sử dụng dễ dàng hơn có thể làm cho các yêu cầu lập trình phức tạp hơn.
    - Tính linh động cao có thể làm cho việc bảo trì nhiều hơn.
    - Thỏa mãn các yêu cầu của người sử dụng này sẽ khó thỏa mãn các yêu cầu của người sử dụng khác.
  - Vấn đề chính là *chất lượng so với chi phí*

Thiết kế và sử dụng mã

- ***Mã*** (*code*) là một tập các chữ và/hoặc số của một cấu trúc vắn tắt dùng cho việc phân loại các mục dữ liệu để lưu trữ, liên lạc, xử lý và lấy dữ liệu.

Thiết kế và sử dụng mã

Lợi ích của mã

- Tạo các danh hiệu duy nhất.
- Cho thấy các đặc tính của các mục dữ liệu.
- Cho thấy các mối liên kết giữa các mục dữ liệu.
- Giảm vùng nhớ và chi phí lưu trữ.
- Giảm thời gian nhập và truyền dữ liệu.
- Thuận tiện cho việc phân loại dữ liệu.
- Thuận tiện cho việc tìm kiếm và lấy dữ liệu.
- Có thể hiển thị hoặc che giấu thông tin.
- Có thể giảm các lỗi sai khi nhập dữ liệu.
- Thao tác dữ liệu một cách hiệu quả: trộn, sắp thứ tự.

Thiết kế và sử dụng mã

- **Các mục đích của mã hóa**
  - **Theo dõi một điều gì đó (*keeping track of something*)**
    - **Mã tuần tự (*sequence code*)**
    - **Mã dẫn xuất (*derivation code*)**
  - **Phân loại thông tin (*classifying information*)**
    - **Mã phân loại (*classification code*)**
    - **Mã tuần tự chia khối (*block sequence code*)**
  - **Che giấu thông tin (*concealing information*)**
    - **Mật mã (*cipher code*)**

Thiết kế và sử dụng mã

- Các mục đích của mã hóa
  - Thể hiện thông tin (*revealing information*)
    - Mã ký số có nghĩa (*significant digit code / faceted code*)
    - Mã viết tắt (*abbreviation code*) / mã gợi nhớ (*mnemonic code*)
  - Yêu cầu một tác vụ thích hợp (*requesting appropriate action*).
    - Mã chức năng (*function code*) / mã tác vụ (*action code*)

Thiết kế và sử dụng mã

Mã tuần tự

- *sequence code*
- Mã tuần tự là mã lấy các giá trị liên tiếp từ tập giá trị mã.
- Mã tuần tự được sử dụng khi thứ tự của việc xử lý cần phải biết chuỗi các mục dữ liệu đưa vào hệ thống hoặc thứ tự của các sự kiện xảy ra.

Empno	Employee Name
001	Smith
002	Adams
003	Tiger

Hình 6.7. *Empno* thuộc loại mã tuần tự.

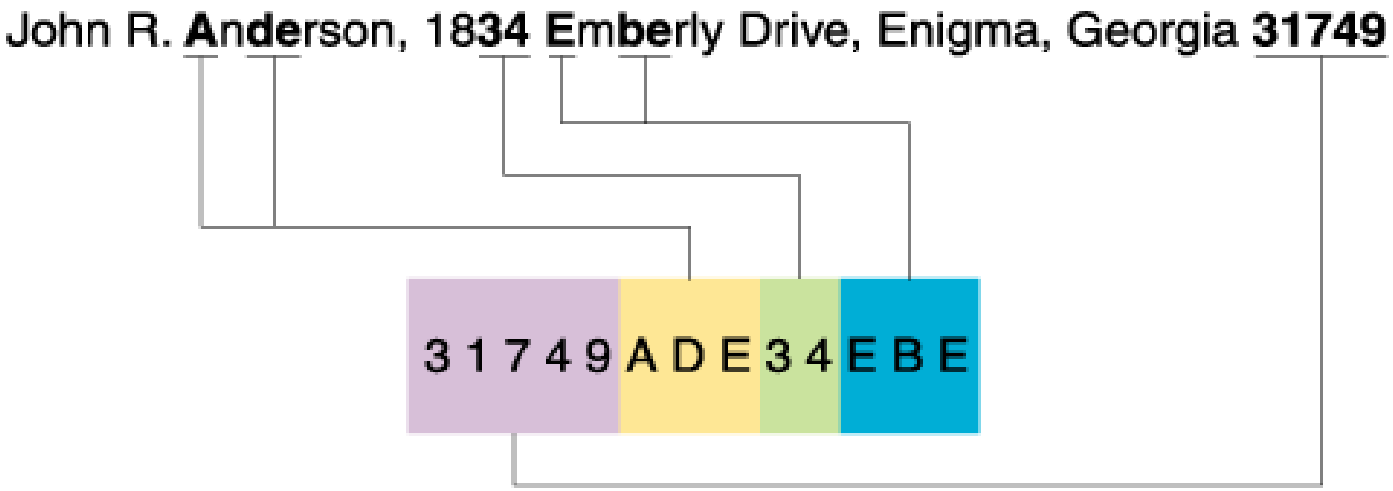
Thiết kế và sử dụng mã

- Mã tuần tự
  - Ưu điểm
    - Đơn giản.
    - Ngắn và duy nhất.
    - Dễ phát sinh giá trị mã liên tiếp.
  - Nhược điểm
    - Không chứa thông tin có ích, ngoại trừ thứ tự trước sau của các giá trị mã.
    - Không linh động.
    - Dễ bị tràn mã vì các mã trống không được sử dụng lại.

Thiết kế và sử dụng mã

❖ Mã dẫn xuất

- *derivation code*
- **Mã dẫn xuất là mã kết hợp dữ liệu của nhiều thuộc tính khác nhau.**



Hình 6.8. Mã của người đặt mua báo được dẫn xuất từ các thành phần khác nhau của tên và địa chỉ.

Thiết kế và sử dụng mã

❖ Mã phân loại

- *classification code*
- Mã phân loại là mã gom nhóm các mục dữ liệu có liên quan với nhau.

Code	Tax Deduction Item
I	INTEREST PAYMENTS
M	MEDICAL PAYMENTS
T	TAXES
C	CONTRIBUTIONS
D	DUES
S	SUPPLIES

Hình 6.9. Gom nhóm các mục khấu trừ thuế bằng cách sử dụng mã phân loại 1-ký tự.

Thiết kế và sử dụng mã

Mã phân loại

Phân loại các mục dữ liệu

- Phân loại các mục dữ liệu là quá trình xác định và gom nhóm các mục dữ liệu thành các loại (lớp) dựa vào các đặc tính chung.
- Định nghĩa các lớp phải thích hợp với mục đích riêng.
- Phân loại các mục dữ liệu nhằm đáp ứng các yêu cầu của người sử dụng.
- Phân loại các mục dữ liệu phải cho phép mở rộng để phân loại các mục dữ liệu mới.
- Các mục dữ liệu phải rõ ràng.

Thiết kế và sử dụng mã

Code	Tax Deductible Item
I	INTEREST PAYMENTS
M	MEDICAL PAYMENTS
T	TAXES
C	CONTRIBUTIONS
D	DUES
S	SUPPLIES
S	SUBSCRIPTIONS
C	COMPUTER
I	INSURANCE
M	MISCELLANEOUS
B	SUBSCRIPTIONS
P	COMPUTER
N	INSURANCE
X	MISCELLANEOUS

Duplicate codes

are corrected by "forcing" the codes to fit

Hình 6.10. Các vấn đề xảy ra cho mã phân loại 1-ký tự khi các loại có cùng ký tự chữ.

Thiết kế và sử dụng mã

❖ Mã tuần tự chia khối

- ***block sequence code***
- **Mã tuần tự chia khối là sự mở rộng của mã tuần tự, phân chia tập giá trị mã thành các nhóm có ý nghĩa khác nhau.**

Thiết kế và sử dụng mã

Range	Field
000 - 099	General Works List
100 - 199	Philosophy, Psychology, Ethics
200 - 299	Religion, Mythology
300 - 399	Social Sciences
400 - 499	Philology
500 - 599	Science
600 - 699	Applied Science
700 - 799	Fine Arts
800 - 899	Literature
900 - 999	History, Geography, Biography, Travel

Hình 6.11. Phân loại thập phân *Dewey* sử dụng mã khối tuần tự.

Thiết kế và sử dụng mã

Code	Name of Software Package	Type
100	1-2-3	Speadsheet
101	Multiplan	
102	VP-Planner	
…	…	
200	R-BASE System V	Database
201	Paradox 1.1	
202	dBASE III Plus	
…	…	
300	Wordstar 4.0	Wordprocessing
301	Wordperfect 4.2	
302	Word 3.1	
…	…	
400	Super Key 1.03	Utilities
401	Pro Key 4.0	
…	…	

Hình 6.12. Sử dụng mã khối tuần tự để gom nhóm các phần mềm tương tự nhau.

Thiết kế và sử dụng mã

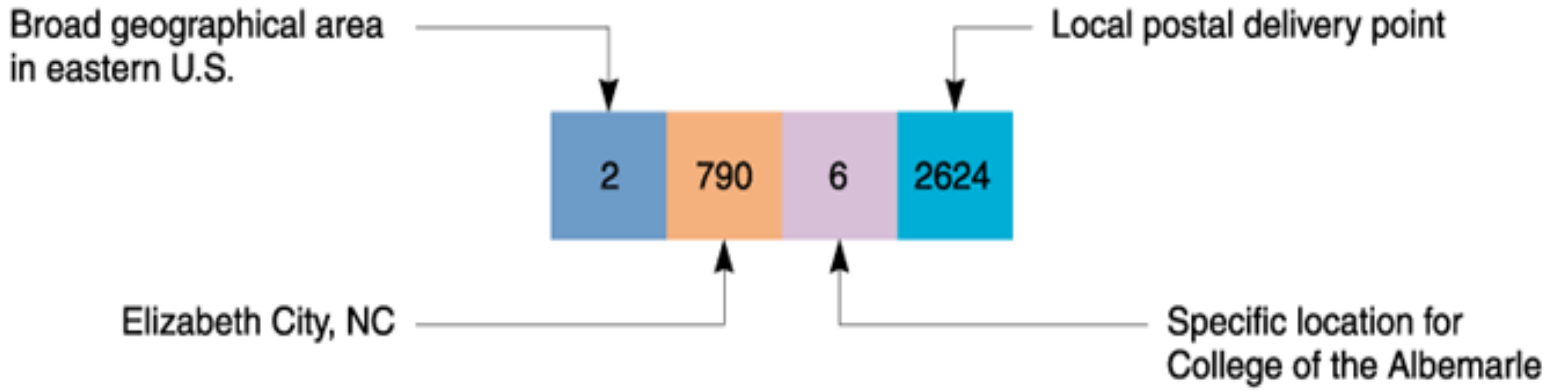
Mật mã

- *cipher code*
- Mật mã là mã mà trong đó một chữ (hay một số) thay thế cho một điều gì đó, hoặc một chữ thay thế cho một số.
- Ví dụ: cửa hàng bán lẻ sử dụng một từ gồm 10 ký tự chữ CAMPGROUND để mã hóa giá bán: C biểu diễn 1, A biểu diễn 2, …, D biểu diễn 0. Do đó, mã GRAND có giá bán là $562.90

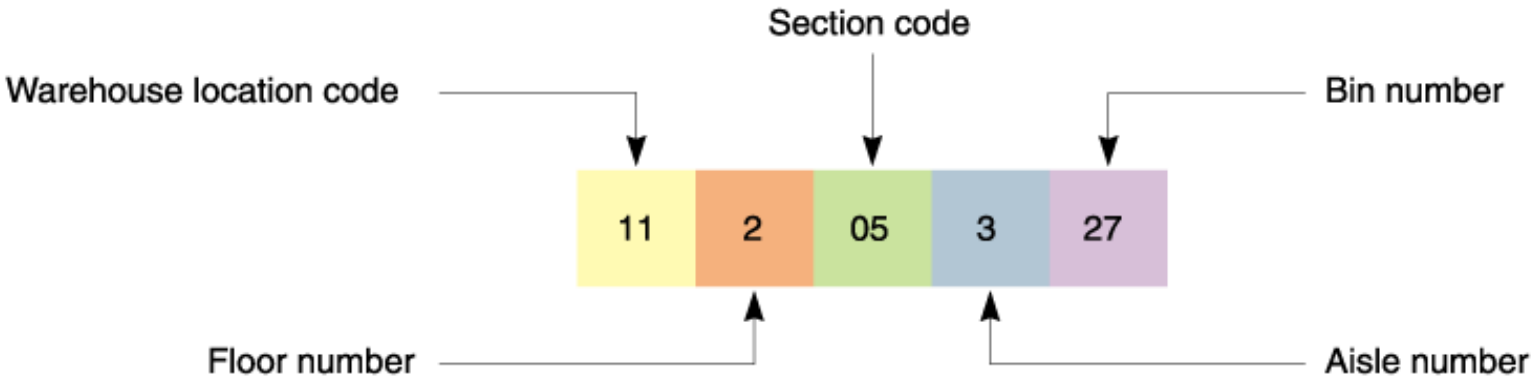
Thiết kế và sử dụng mã

- **Mã khối**
  - *block code*
  - Mã khối là mã được phân chia thành các khối mà mỗi khối có sự phân loại riêng.
  - Mã ký số có nghĩa (*significant digit code / faceted code*)
    - Các nhóm ký số riêng biệt biểu diễn các đặc tính của mục dữ liệu.
    - Xác định số ký số cho mỗi đặc tính.
  - Mã khối phân cấp (*hierarchical block code*)
    - Sắp xếp theo thứ tự quan trọng của các khối.
    - Khối bên trái là lớp cha, có ý nghĩa tổng quát.
    - Khối bên phải là lớp con, có ý nghĩa chi tiết.

Thiết kế và sử dụng mã



Hình 6.13. Mã vùng điện thoại (*ZIP code*) là một ví dụ về mã ký số có nghĩa sử dụng các nhóm con để cho biết thông tin.



Hình 6.14. Mã sử dụng các ký số có nghĩa để xác định vị trí của mặt hàng tồn kho.

Thiết kế và sử dụng mã

Mã khối

Ưu điểm

- Giá trị và vị trí đều có ý nghĩa.
- Cấu trúc mã thuận tiện cho việc xử lý thông tin, dễ dàng sắp thứ tự, phân tích, thao tác và lấy các mục dữ liệu.
- Dễ dàng mở rộng sự phân loại vì mỗi khối có sự phân loại riêng, ngoại trừ giá trị lớn nhất của khối đã được sử dụng.

Nhược điểm

- Chiều dài của mã phụ thuộc vào số lượng các thuộc tính được phân loại, thông thường là mã dài.
- Mã có thể chứa những số dự phòng.
- Nếu khóa của mẫu tin là mã khối thì cần phải giải quyết các vấn đề duy trì hệ thống mã khi muốn phải thay đổi khóa.

Thiết kế và sử dụng mã

❖ Mã gợi nhớ

- *mnemonic code*
- Mã gợi nhớ là mã sử dụng các ký tự số và/hoặc ký tự chữ để mô tả mục dữ liệu.
- Giá trị mã có trợ giúp gợi nhớ.
- Chỉ sử dụng mã gợi nhớ khi số lượng các mục dữ liệu không nhiều (dưới 50 mục dữ liệu).

Code	City
SIN	Singapore
NYK	New York
HKG	Hongkong
LON	London

Hình 6.15. Mã gợi nhớ được sử dụng trong *International Air Transport Association* (IATA).

Thiết kế và sử dụng mã

Mã tác vụ

- *action code*
- Mã tác vụ là mã cho biết tác vụ được thực hiện tương ứng với mục dữ liệu.

Code	Action
1	Delivered
2	Sold
3	Spoiled
4	Lost or stolen
5	Returned
6	Transferred out
7	Transferred in
8	Journal entry (add)
9	Journal entry (subtract)

Hình 6.16. Một ví dụ về mã tác vụ.

Thiết kế và sử dụng mã

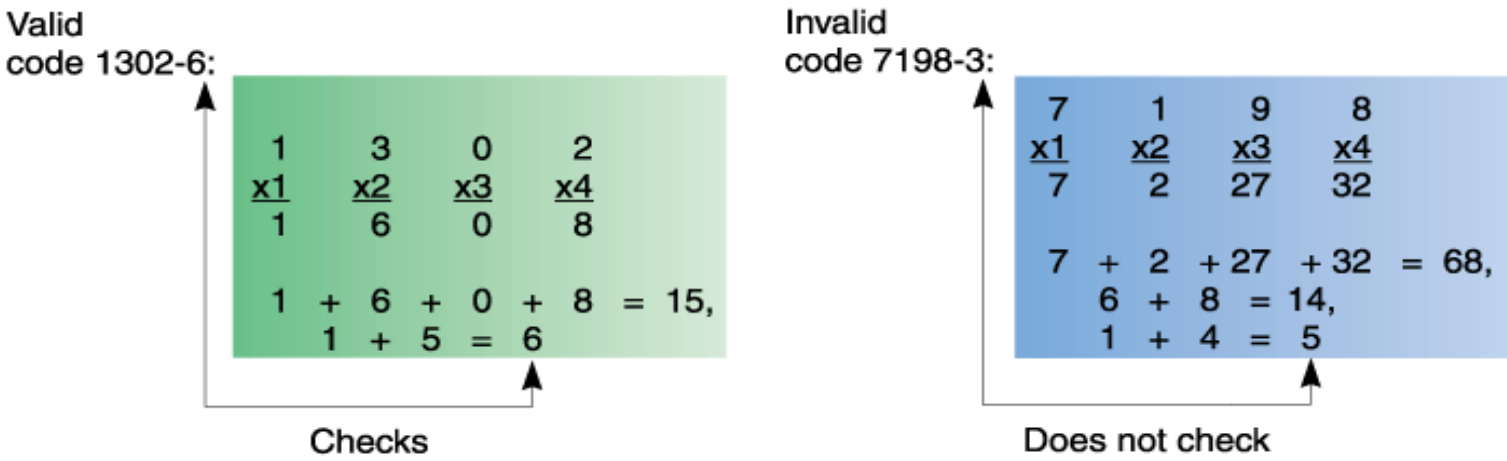
Mã tự kiểm tra

- *self-checking code*
- Mã tự kiểm tra là mã có chứa các ký số được xác định từ các ký số còn lại để xác định sự hợp lệ của mã.
- Mã tự kiểm tra đề phòng các lỗi sai tiêu biểu:
  - Lỗi sai do sao chép lại (*transcription error*): một ký số bị sai.
  - Lỗi sai do đổi chỗ (*transposition error*): đổi chỗ 2 ký số liên tiếp.
  - Lỗi sai do đổi chỗ kép (*double tranposition error*): đổi chỗ 2 ký số cách nhau một ký số.
  - Lỗi sai ngẫu nhiên (*random error*): nhiều ký số bị sai.

Thiết kế và sử dụng mã

Error type	Invalid	Valid
Transcription error	12845	12345
Transposition error	783156	781356
Double transposition error	301157	305117
Random error	987866	987680

Hình 6.17. Các loại lỗi sai.



Hình 6.18. Mã 1302-6 có ký số kiểm tra hợp lệ. Mã 7198-3 có ký số kiểm tra không hợp lệ.

Thiết kế và sử dụng mã

Mã tự kiểm tra

Các bước tạo các ký số kiểm tra (*check digit*)

- Mã ban đầu $X_1X_2 \ldots X_n$.
- Chọn các trọng số $W_1W_2 \ldots W_n$
  - arithmetic progression: 6 5 4 3 2
  - geometric progression: 32 16 8 4 2
  - prime number: 17 13 7 5 3
- Tính $S = \sum X_iW_i$
- Tính R = S mod M, với M là một số nguyên tố
- Tính C = M - R. Nếu R = 0 thì C = 0.

Thiết kế và sử dụng mã

Mã ban đầu: 12345 M = 11
Arithmetic progression $S = 1 \times 6 + 2 \times 5 + 3 \times 4 + 4 \times 3 + 5 \times 2 = 50$ $R = S \bmod M = 50 \bmod 11 = 6$ $C = M - R = 11 - 6 = 5$ Mã cuối cùng: 12345 - 5
Geometric progression $S = 1 \times 32 + 2 \times 16 + 3 \times 8 + 4 \times 4 + 5 \times 2 = 114$ $R = S \bmod M = 114 \bmod 11 = 4$ $C = M - R = 11 - 4 = 7$ Mã cuối cùng: 12345 - 7
Prime number $S = 1 \times 17 + 2 \times 13 + 3 \times 7 + 4 \times 5 + 5 \times 3 = 99$ $R = S \bmod M = 99 \bmod 11 = 0$ $C = 0$ Mã cuối cùng: 12345 - 0

Hình 6.19. Ví dụ về các cách khác nhau để tạo ký số kiểm tra.

Thiết kế và sử dụng mã

- **Quá trình thiết kế mã**
  - Xác định các mục dữ liệu cần được mã hóa.
  - Xác định khối lượng dữ liệu.
  - Xác định các mục tiêu của mã.
  - Xác định phạm vi sử dụng mã.
  - Xác định chu kỳ thời gian sử dụng mã.
  - Xác định loại mã.

Thiết kế và sử dụng mã

❖ Một số điều lưu ý khi thiết kế mã

- Cấu trúc mã phải thỏa mãn các yêu cầu của người sử dụng và phải có phương pháp xử lý mã một cách thích hợp.
- Mỗi mã phải có cấu trúc biểu diễn duy nhất.
- Thiết kế mã phải linh động để đáp ứng việc thay đổi các yêu cầu.
- Cấu trúc mã phải dễ hiểu (đơn giản, thực tế, có ý nghĩa).
- Cần phải chuẩn hóa các mục dữ liệu được mã hóa.

Thiết kế và sử dụng mã

- **Chuẩn hóa các mục dữ liệu được mã hóa**
  - Tránh các ký tự mà chữ viết hoặc phát âm gần giống nhau (ví dụ: O, Z, I, S, V có thể nhầm với 0, 2, 1, 5, U).
  - Tránh các lỗ trống (các giá trị mã không được sử dụng) trong tập giá trị mã.
  - Các ngày và các tuần cần được đánh số (ngày từ 1 đến 7, tuần từ 1 đến 52).
  - Sử dụng dạng 24 giờ thay vì dạng AM/PM (ví dụ: 13:00 thay vì 01:00 PM).
  - Sử dụng dạng ngày biểu diễn bằng các ký số thay vì bằng ký tự chữ (ví dụ: 18/09/2004 thay vì 18th Sept. 2004).

Thiết kế và sử dụng mã

Chiều dài của mã

Mã ngắn (*short code*)

- Mã có ít ký tự.
- Dễ dàng phân loại, tạo một giá trị mã mới.
- Tốn ít vùng nhớ lưu trữ.
- Tốn ít thời gian truyền.

Mã dài (*long code*)

- Mã có nhiều ký tự.
- Mã có thể chứa một số thông tin cần thiết.
- Dễ dàng cho việc lấy dữ liệu.
- Dễ dàng cho việc xử lý thông tin.

Thiết kế và sử dụng mã

- **Sử dụng mã**
  - Duy trì mã ngắn gọn.
  - Cho phép mở rộng mã.
  - Duy trì mã ổn định.
  - Tạo mã duy nhất.
  - Sử dụng mã có thể sắp thứ tự được.
  - Tránh nhầm lẫn mã.
  - Tạo mã có ý nghĩa.
  - Sử dụng mã cho mục đích riêng biệt.
  - Duy trì các mã nhất quán với nhau.

Thiết kế và sử dụng mã

Incorrect Sorting Using MMM-DD-YYY	Incorrect Sorting Using MM-DD-YYYY	Incorrect Sorting (Year 2000 Problem) YY-MM-DD	Correct Sorting Using YYYY-MM-DD
Dec-25-1998	06-04-1998	00-06-11	1997-06-12
Dec-31-1997	06-11-2000	97-06-12	1997-12-31
Jul-04-1999	06-12-1997	97-12-31	1998-06-04
Jun-04-1998	07-04-1999	98-06-04	1998-10-24
Jun-11-2000	10-24-1998	98-10-24	1998-12-25
Jun-12-1997	12-25-1998	98-12-25	1999-07-04
Oct-24-1998	12-31-1997	99-07-04	2000-06-11

Hình 6.20. Các vấn đề sắp thứ tự dữ liệu kiểu *Date*.

Thiết kế và sử dụng mã

<table>
<tr><th colspan="6">CODE DESIGN FORM</th></tr>
<tr><td colspan="3">Document Reference Number:</td><td colspan="3">Code Design Form Number:</td></tr>
<tr><td colspan="2">Name of coded item:</td><td>Numbering method:</td><td>Number of columns:</td><td colspan="2">Check digit (Yes/No):</td></tr>
<tr><td colspan="2">Number of code numbers assigned</td><td rowspan="2">Period of use:</td><td colspan="3" rowspan="2">Range of use:</td></tr>
<tr><td>Current:</td><td>Future:</td></tr>
<tr><td colspan="6">Coding objectives:</td></tr>
<tr><td colspan="6">Structure:</td></tr>
<tr><td colspan="6">Range of numbers:</td></tr>
<tr><td colspan="6">Special properties:</td></tr>
</table>

Hình 6.21. Mẫu thiết kế mã.