

## 7. INTERFACE DESIGN



## Interface design

- ➡ 1. Graphical user interface design
2. System/Device interface design

## References

- [1] Textbook for Software Design & Development Engineers, No. 3 – *System Development, Operations and Maintenance*, 2<sup>nd</sup> Edition; Japan Information Processing Development Corporation, Japan Information-Technology Engineers Examination Center.

## 1. Graphical user interface design

- ➡ 1.1. Standardizing the screen configuration
- 1.2. Creating screen images
- 1.3. Creating a screen transition diagram
- 1.4. Creating screen specifications

## Standardizing

### ◆ Display

- Physical size, resolution, and number of colors supported by displays

### ◆ Screen: divided into displayed objects called windows (Window)

- Location of standard buttons (e.g., OK, Cancel, Register, Search)
- Display location of messages, etc.
- Display of screen title and menus
- Consistency in expression of alphanumeric characters
- Expression of sentences and detailed items
- Color coordination

## Standardizing

### • Control

- Style, size, color, and characters displayed
- Input check process
- Sequence of moving the focus (e.g., defining the tab sequence)

### • Menu

- Design menus with consideration of the standard specification (common client area) of the screen

### • Direct input from a keyboard

- Maintain consistency in the assignment of shortcut keys

## Standardizing

### • Messages

- Determine how messages are displayed when a time-consuming process is executed (busy).

### • Error

- Execute standardized processing if an error occurs

### • Help

- Develop detailed Help information in accordance with the manual, and maintain consistency in terminology, descriptions, and explanations of methods.

## 1. Graphical user interface design

### 1.1. Standardizing the screen configuration

### ➡ 1.2. Creating screen images

### 1.3. Creating a screen transition diagram

### 1.4. Creating screen specifications

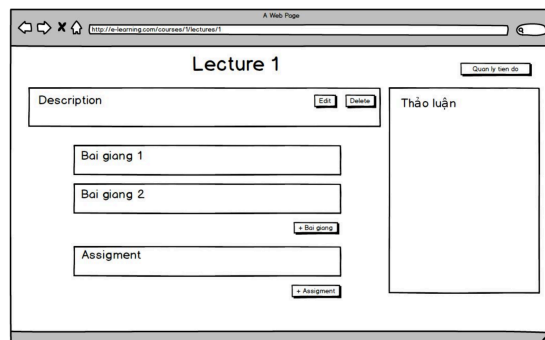
## From use case

- Based on use case and boundary classes which interact with users
  - Map these boundary classes to screens
- Based on input/output description in use case specification/scenario

=> Design screen using tools

## GUI Design tools

- Simple tools
  - Notepad
  - Microsoft Excel/Powerpoint/Word/FrontPage
- Professional tools
  - Free
    - InVision
    - IDEs: Eclipse, NetBeans
  - Commercial
    - Adobe Dreamweaver
    - Axure RP
    - Photoshop
    - IDEs: Visual Studio



## 1. Graphical user interface design

- 1.1. Standardizing the screen configuration
- 1.2. Creating screen images
- 1.3. Creating a screen transition diagram
- 1.4. Creating screen specifications

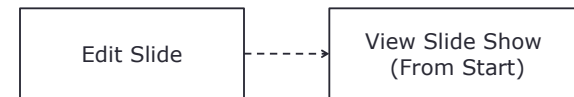
## Display transition diagram

- Summarize the correlation of screens in the screen transition diagram
- Classify the screens into the four patterns by focusing on the transition pattern
- Link the screens in accordance with the classifications

## Four transition patterns

### ◆ 1. Simple screen transition:

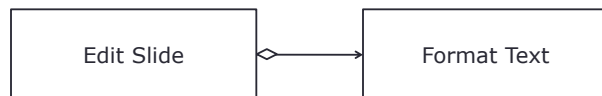
- A conventional simple transition to an independent screen



## Four transition patterns (2)

### • 2. Transition to a dependent child screen:

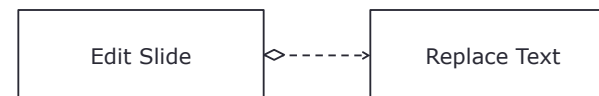
- Move to a pop-up screen
- When a child screen is displayed on the parent screen, the underlying parent screen cannot be operated

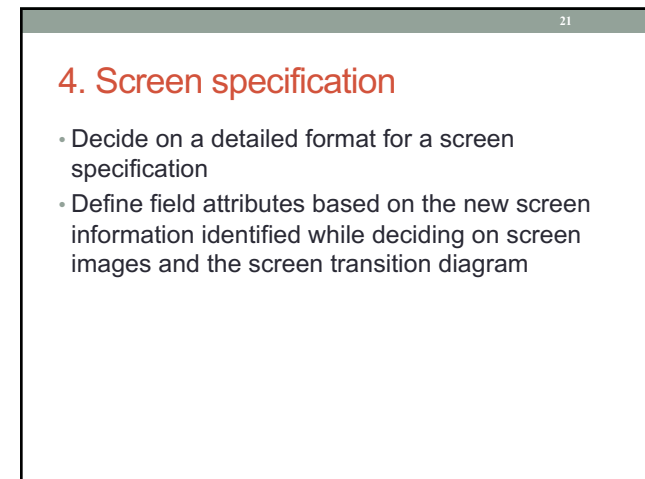
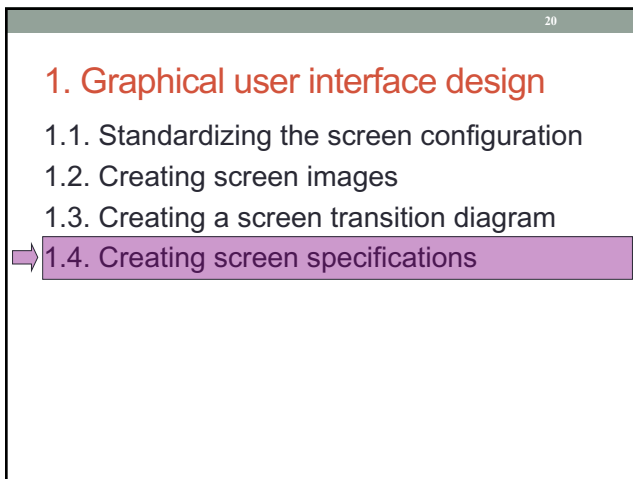
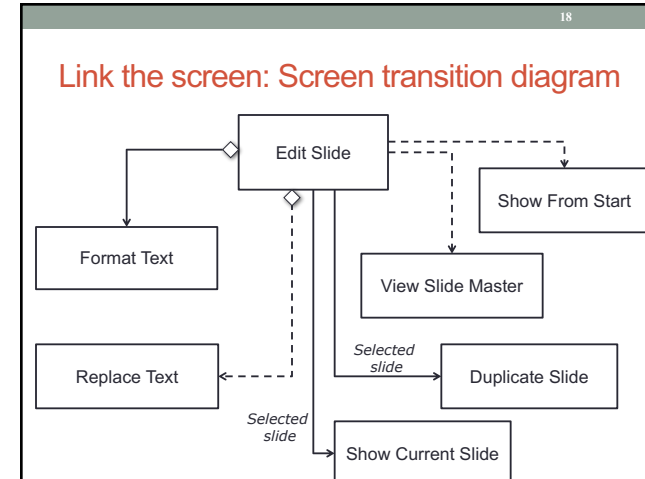
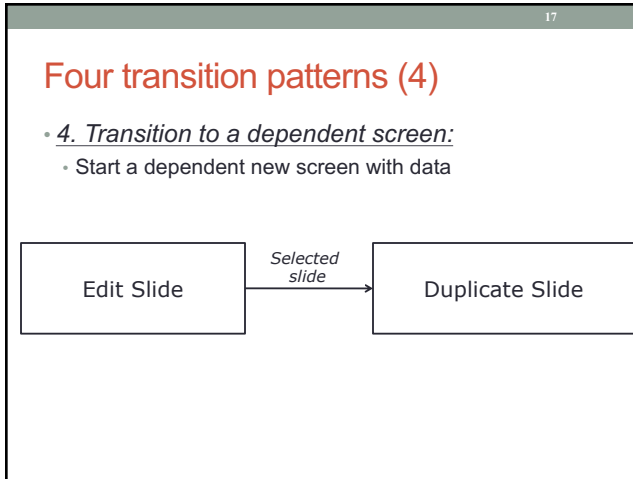


## Four transition patterns (3)

### • 3. Transition to an independent child screen:


- Move to a pop-up screen,
- Parent screen and other screens can be operated while the child screen is displayed.





## Screen specification

- Screen image
  - This is the screen image to be displayed. If screen images are created in advance with the screen design tool, attach a hardcopy.
- List of functions
  - Defines the names of parts such as the buttons on the screen, and summarizes their functions.
  - Provide descriptions of events for individual screens, attributes of parts, input check specifications and output specifications, etc.
- Defining the field attributes

Liquor sales basic system (general-purpose search subsystem for sales information)		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Displaying detail table				
<b>Screen specification example</b>  		Control	Operation	Function	
		Area for displaying detail table	Initial	-Displays in a table information meeting the conditions defined in the search specification screen. -This follows the setting specified in the display settings screen for display items and sequence of display.	
		Graph display button	Click	Displays the graph display screen	
		Table print button	Click	Displays the print preview screen	
[1]: Section 3.2.1, pp 3-54		Return button	Click	Displays the search specification screen	

## Defining the field attributes

- Decide on the field attributes of input and output items
- Summarize them in descriptions of items for screen display.
- The screen consists of multiple fields.
- Each field consists of a one-byte (equivalent to a single character) attribute at the beginning and a variable item

### Example: Defining the field attributes

Screen name	Order entry		[1]		
Item name	Number of digits (bytes)	Type	Field attribute	Remarks	
Transaction category	3	Numeral	Green (blink)	Error items blink.	
Customer code	5	Numeral	Green (blink)	Error items blink.	
Customer name	30	Character	White	15 characters, left-justified	
Product code	8	Numeral	Green (blink)	Error items blink.	
Product name	22	Character	White	11 characters, left-justified	
Quantity	6	Numeral	Green (blink)	Error items blink.	
Unit price	7	Numeral	White		
Amount	9	Numeral	White		
Quantity in stock	10	Numeral, special character	White	Displayed in the format of ZZZ, ZZZ, ZZ9	

[1]: Section 3.2.1, pp 3-57

## Interface design

1. Graphical user interface design

➡ 2. System/Device interface design

## 2. System/Device interface design

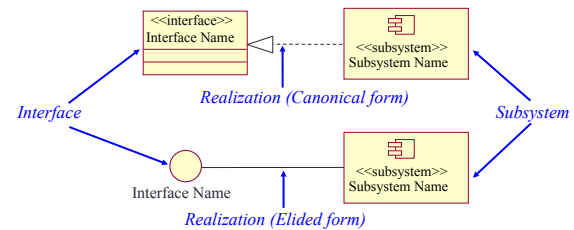
➡ 2.1. Identify subsystem

2.2. Identify subsystem interfaces

2.3. Subsystem design

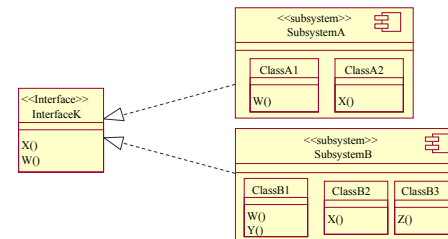
## Review: Subsystems and Interfaces

- Realizes one or more interfaces that define its behavior



## Subsystems and Interfaces (continued)

- Subsystems :
  - Completely encapsulate behavior
  - Represent an independent capability with clear interfaces (potential for reuse)
  - Model multiple implementation variants



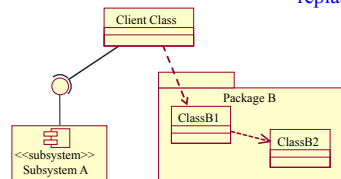
## Packages versus Subsystems

### Subsystems

- Provide behavior
- Completely encapsulate their contents
- Are easily replaced

### Packages

- Don't provide behavior
- Don't completely encapsulate their contents
- May not be easily replaced



Encapsulation is the key!

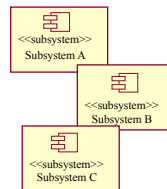
## Subsystem Usage

- Subsystems can be used to partition the system into parts that can be independently:
  - ordered, configured, or delivered
  - developed, as long as the interfaces remain unchanged
  - deployed across a set of distributed computational nodes
  - changed without breaking other parts of the systems
- Subsystems can also be used to:
  - partition the system into units which can provide restricted security over key resources
  - represent existing products or external systems in the design (e.g. components)

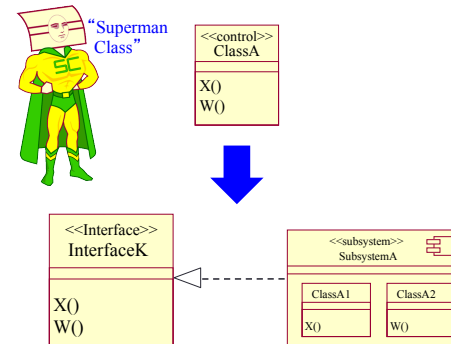
Subsystems raise the level of abstraction.

## Candidate Subsystems

- Analysis classes which may evolve into subsystems:
  - Classes providing complex services and/or utilities
  - Boundary classes (user interfaces and external system interfaces)
- Existing products or external systems in the design (e.g., components):
  - Communication software
  - Database access support
  - Types and data structures
  - Common utilities
  - Application-specific products



## Identifying Subsystems





## 2. System/Device interface design

### 2.1. Identify subsystem

### 2.2. Identify subsystem interfaces

### 2.3. Subsystem design

## Identifying Interfaces

- Purpose
  - To identify the interfaces of the subsystems based on their responsibilities
- Steps
  - Identify a set of candidate interfaces for all subsystems.
  - Look for similarities between interfaces.
  - Define interface dependencies.
  - Map the interfaces to subsystems.
  - Define the behavior specified by the interfaces.
  - Package the interfaces.

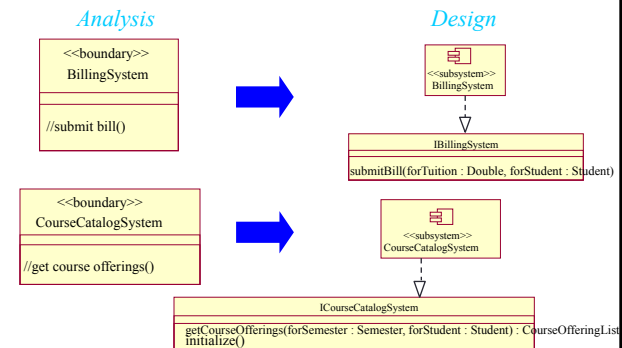
Stable, well-defined interfaces are key to a stable, resilient architecture.

## Interface Guidelines

- Interface name
  - Reflects role in system
- Interface description
  - Conveys responsibilities
- Operation definition
  - Name should reflect operation result
  - Describes what operation does, all parameters and result
- Interface documentation
  - Package supporting info: sequence and state diagrams, test plans, etc.



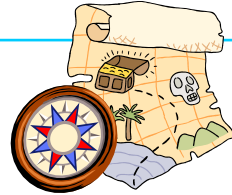
## Example: Design Subsystems and Interfaces



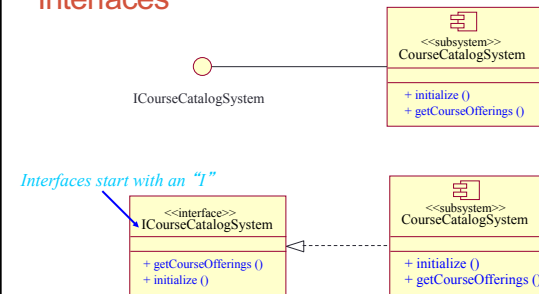
All other analysis classes map directly to design classes.

### Example: Analysis-Class-To-Design-Element Map

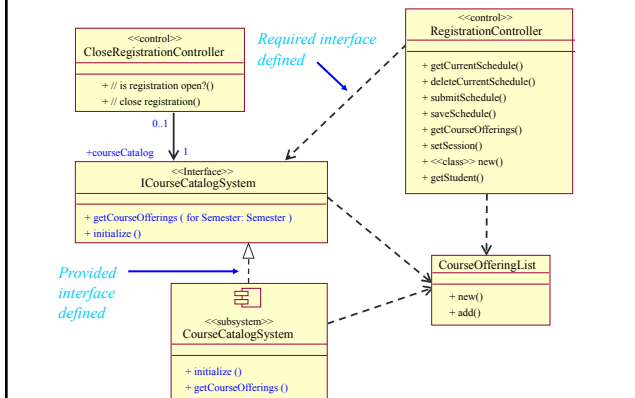
Analysis Class	Design Element
CourseCatalogSystem	CourseCatalogSystem Subsystem
BillingSystem	BillingSystem Subsystem
All other analysis classes map directly to design classes	



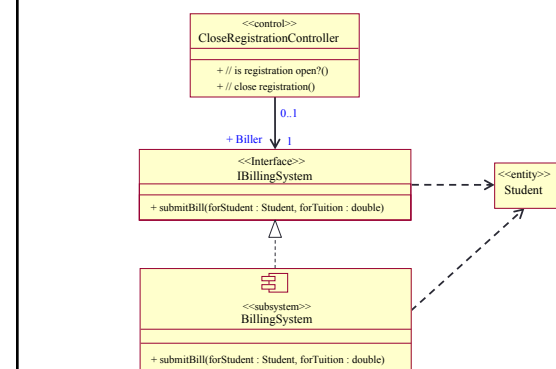
### Modeling Convention: Subsystems and Interfaces



### Example: Subsystem Context: CourseCatalogSystem



### Example: Subsystem Context: Billing System



## 2. System/Device interface design

- 2.1. Identify subsystem
- 2.2. Identify subsystem interfaces
- ➔ 2.3. Subsystem design

## Subsystem design

- Refer to 12-Subsystem Design (IBM).pdf

## Question?



## Bài tập tuần

- Tất cả các use case
  - Tìm các lớp phân tích cho từng use case
  - Gom nhóm thành các package, giải thích lý do
  - Vẽ biểu đồ lớp (không cần thuộc tính, hành vi) cho từng package, vẽ biểu đồ phụ thuộc giữa các package
    - Chú ý với các use case có người phụ trách cần có hành vi cho lớp phân tích
  - Vẽ biểu đồ dịch chuyển màn hình của các lớp GUI
- Với 2 use case của cá nhân
  - Biểu đồ tương tác (1 trong 2 hoặc cả hai loại: Trình tự / Giao tiếp)
  - Biểu đồ lớp phân tích cho từng use case (có hành vi)
  - Thiết kế, đặc tả màn hình cho các lớp GUI