Systems Analysis and Design with UML Version 2.0, Second Edition

Alan Dennis, Barbara Wixom, and David Tegarden

Chapter 9: Moving on to Design

John Wiley & Sons, Inc.

WILEY

Copyright © 2005 John Wiley & Sons, Inc.

WILEY

Moving On To Design

Chapter 9

Key Ideas

- The purpose of the analysis phase is to figure out what the business needs. The purpose of the design phase is to figure out *how to provide it*.
- The steps in both analysis and design phases are highly *interrelated* and may require much "going back and forth"

Objectives

- Understand the transition from analysis to design.
- Understand the use of factoring, partitions, and layers.
- Be able to create package diagrams.
- Be familiar with the custom, packaged, and outsource design alternatives.
- Be able to create an alternative matrix.

REVISITING THE OBJECT-ORIENTED APPROACH TO ANALYSIS AND DESIGN

OO Analysis and Design Foundation

- Use-case driven
- Architecture centric
- Iterative and incremental

Combining Three Views

- Functional
- Static
- Dynamic

A "Minimalist" Approach

- Planning
- Gathering requirements
- Perform a series of "builds"
- Use results of each build as feedback for design and implementation

EVOLVING THE ANALYSIS MODELS INTO DESIGN MODELS

Avoid Classic Design Mistakes

- Reducing design time
- Feature creep
- Silver bullet syndrome
- Switching tools in mid-project

Factoring

- Creating modules that account for similarities and differences between units of interest
- New classes
  - Generalization
  - Aggregation
- Abstracting
- Refinement

Partitions and Collaborations

- Creating "subsystems" or larger units
- Grouping units that collaborate
- May have collaboration among units or partitions
- The more messages or contracts between objects, the more likely they are in the same partition

Purpose of Layers

- Model-view-controller (MVC) architecture
  - Models implement application logic
  - Views and controllers do user interfaces
- Separating application logic from user interface logic

Layers

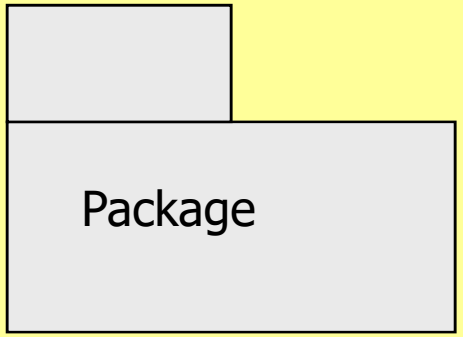
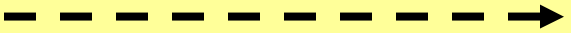
Layers	Examples	Relevant Chapters
Foundation	Date, Enumeration	9, 10
Physical Architecture	ServerSocket, URLConnection	10, 13
Human Computer Interaction	Button, Panel	10, 12
Data Management	DataInputStream, FileInputStream	10, 11
Problem Domain	Employee, Customer	6, 7, 8, 9, 10

PACKAGES AND PACKAGE DIAGRAMS

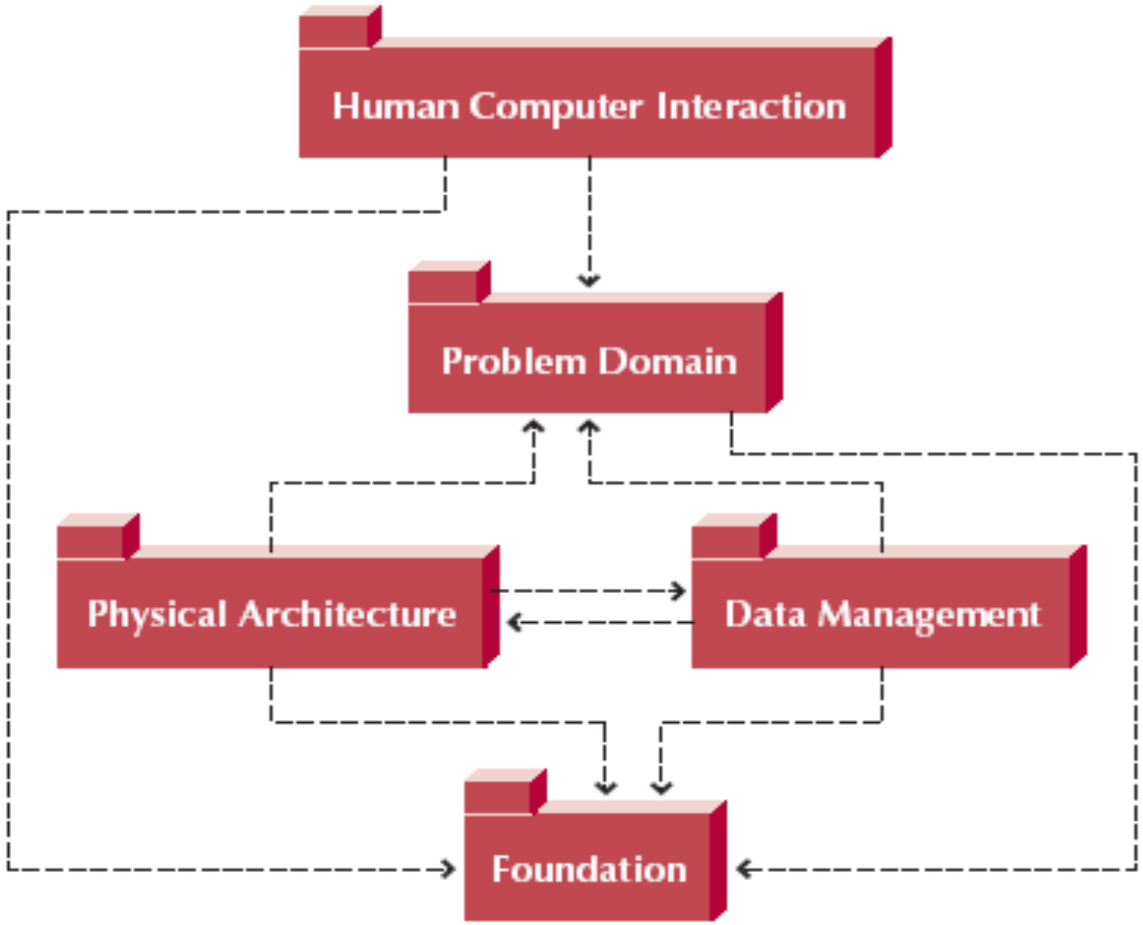
Packages

- Logical grouping of UML elements
- Simplifies UML diagrams
  - Groups related elements into a single higher-level element
- Dependency relationships
  - Shows a dependency between packages

Syntax for Package Diagram

A PACKAGE	
A DEPENDENCY RELATIONSHIP	

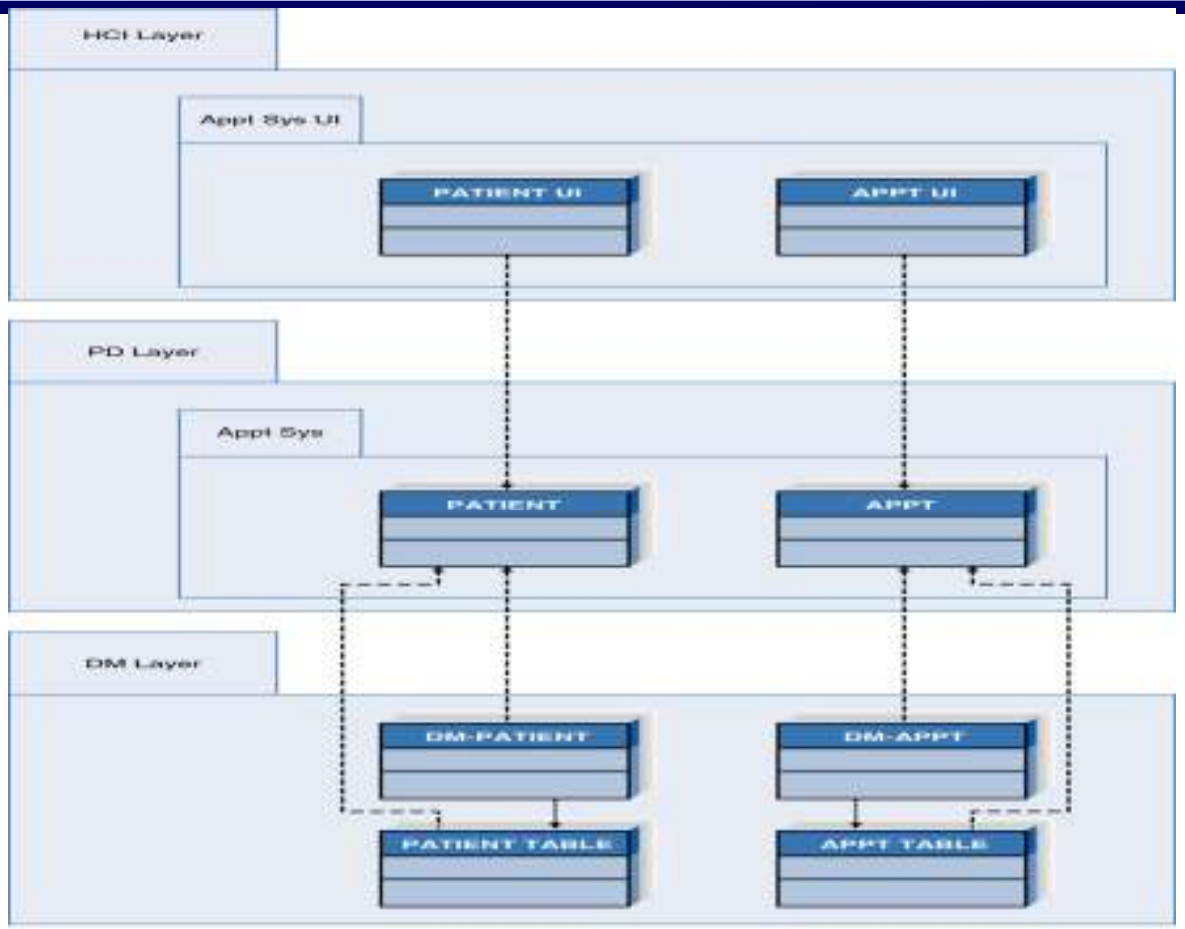
Package Diagram of Dependency Relationships Among Layers



Modification Dependency

- Indicates that a change in one package could cause a change to be required in another package.
- Example:
  - A change in one method will cause the interface for all objects of this class to change. Therefore, all classes that have objects that send messages to the instances of the modified class could have to be modified.

Package Diagram of Appointment System



Dennis: SAD
Fig: 8-6 W-42 100% of size
Fine Line Illustrations (516) 501-0400

Steps for Identifying Packages and Building Package Diagrams

- Set the context
- Cluster classes together based on shared relationships
- Model clustered classes as a package
- Identify dependency relationships among packages
- Place dependency relationships between packages

Package Diagram of the PD Layer for the Appointment System

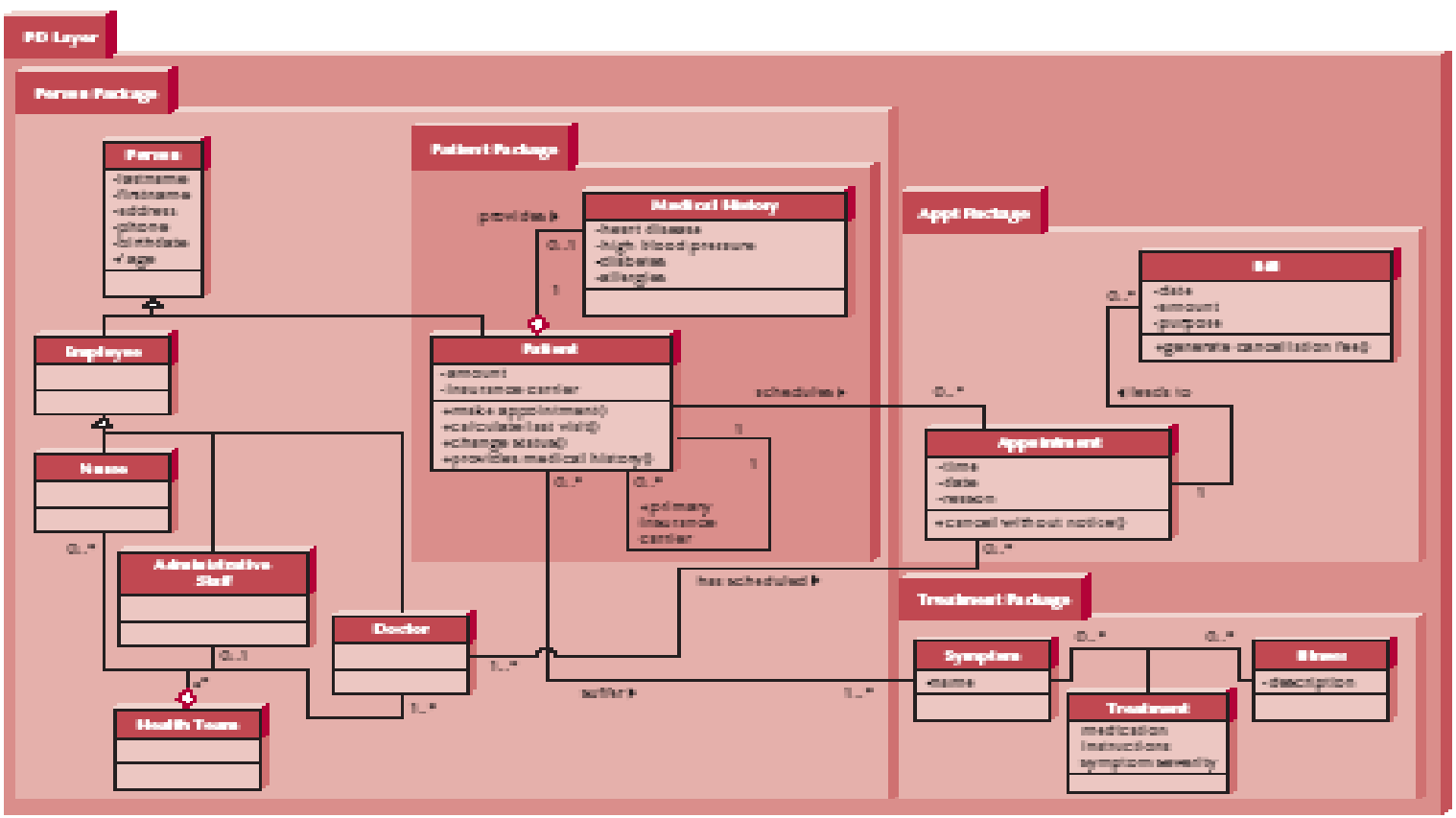


FIGURE 9-6 Package Diagram of the PD Layer for the Appointment System

CD Selections

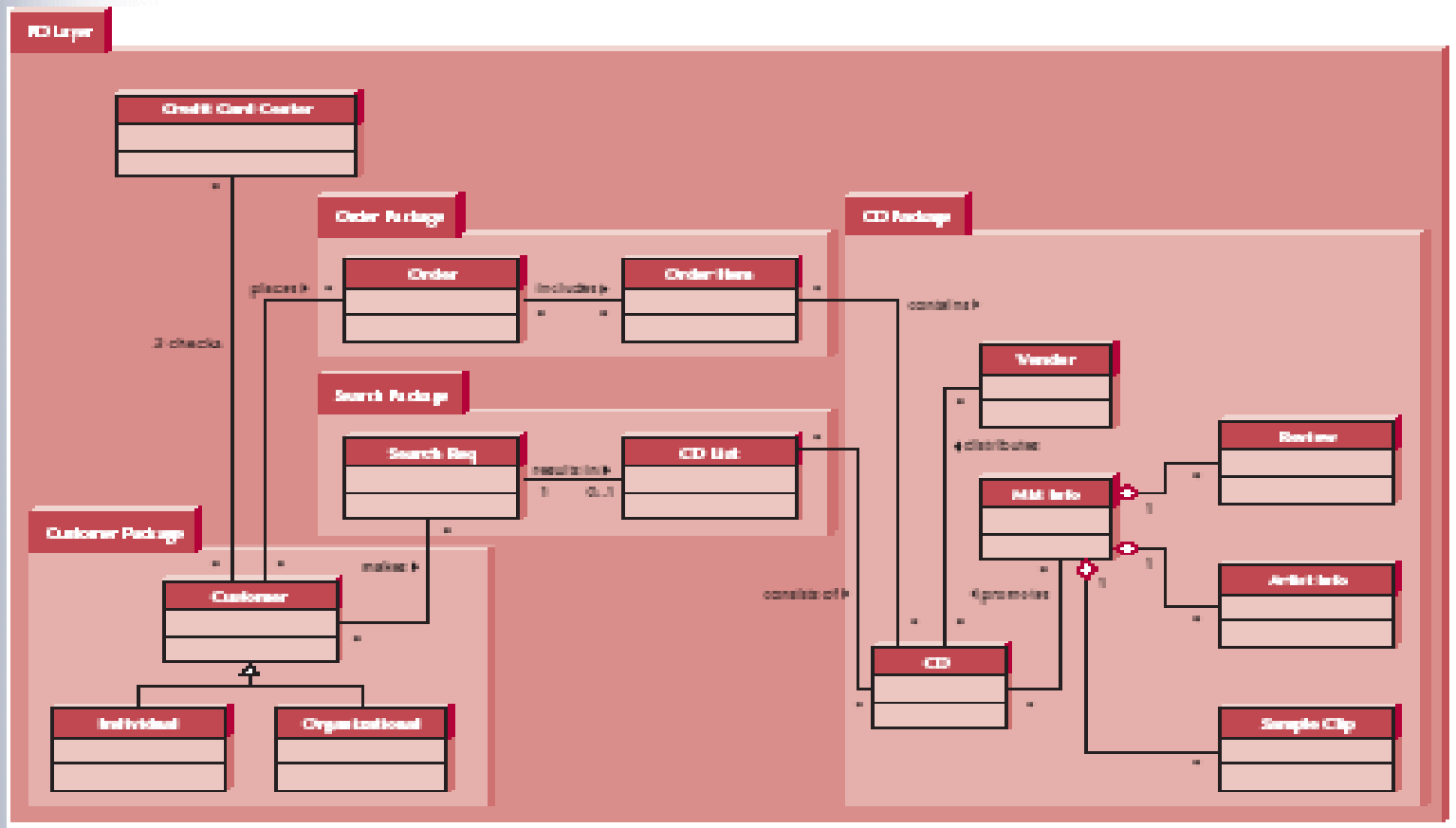


FIGURE 9-8 Package Diagram of the PD Layer of CD Selections Internet Sales System

CD Selections

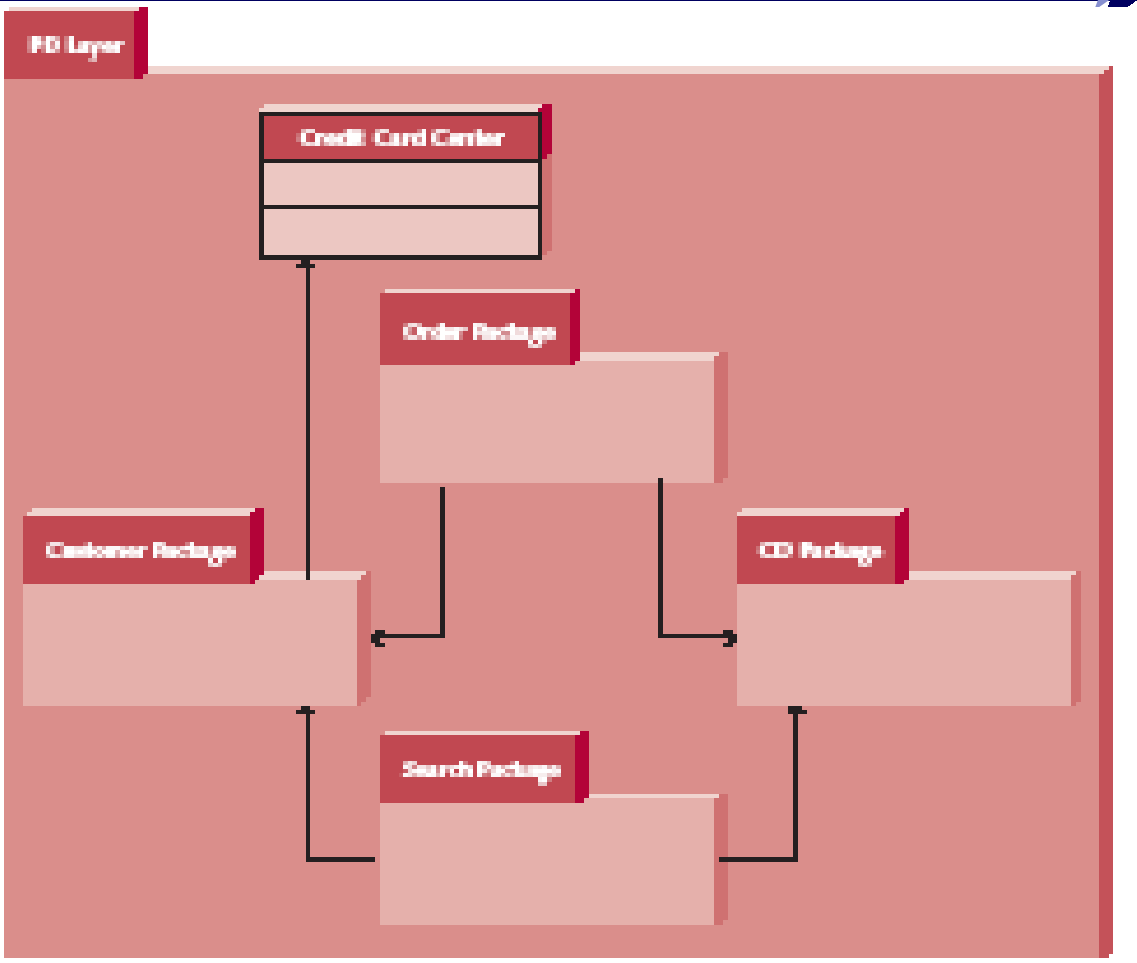


FIGURE 9-9
Overview Package Diagram of the PD Layer of CD Selections Internet Sales System

DESIGN STRATEGIES

Custom Development

- Allows for meeting highly specialized requirements
- Allows flexibility and creativity in solving problems
- Easier to change components
- Builds personnel skills
- May tax firm's resources
- May add significant risk

Packaged Software

- Software already written
- May be more efficient
- May be more thoroughly tested and proven
- May range from components to tools to whole enterprise systems
- Must accept functionality provided
- May require change in how the firm does business
- May require significant “customization” or “workarounds”

System Integration

- The process of combining packages, legacy systems, and new software
- Key challenge is integrating data
- Write data in the same format
- Revise existing data formats
- Develop “object wrappers”

Outsourcing

- Hire external firm to create system
- May have more skills
- May extend existing resources
- Never outsource what you don't understand
- Carefully choose vendor
- Prepare contract and payment style carefully

Outsourcing Guidelines

- Keep lines of communication open with outsourcer
- Define and stabilize requirements before signing a contract
- View outsourcing relationship as partnership
- Select outsource vender carefully
- Assign person to manage relationship
- Don’t outsource what you don’t understand
- Emphasize flexible requirements, long-term relationships, and short-term contracts

Selecting a Design Strategy

- Business need
- In-house experience
- Project skills
- Project management
- Time frame

Your Turn

- Suppose that your university is interested in creating a new course registration system that can support Web-based registration?
- What should the university consider when determining whether to invest in a custom, packaged, or outsourcing system solution?

DEVELOPING THE ACTUAL DESIGN

The Alternative matrix

- Combines several feasibility analyses into one grid
- Revisits technical, economic, and organizational feasibility

Request for Proposals

- Description of the system you propose to be built
- Vendors, developers, service providers respond with proposals including how they will address needs as well as stating cost and time requirements.

CD Selections

	Alternative 1: Shop With Me	Alternative 2: WebShop	Alternative 3: Shop-N-Go
Technical feasibility	Developed using C; very little C experience in-house Orders sent to company using e-mail files	Developed using C and Java; would like to develop in-house Java skills Flexible export features for passing order information to other systems	Developed using Java; would like to develop in-house Java skills Orders saved to a number of file formats
Economic feasibility	$150 initial charge	$700 up-front charge, no yearly fees	$200/year
Organizational feasibility	Program used by other retail music companies	Program used by other retail music companies	Brand-new application; few companies have experience with Shop-N-Go to date
Other benefits	Very simple to use	Tom in Information Systems support has had limited but positive experience with this program Easy to customize	
Other limitations			The interface is not easily customized

Summary

- When evolving analysis into design models, it is important to review the analysis models then add system environment information.
- *Packages and package diagrams* help provide structure and less complex views of the new system.
- *Custom building, packages, and outsourcing* are alternative ways of creating the new system.
- The alternative matrix can help with the selection of a design strategy.

Expanding the Domain

- Smalltalk is an object-oriented programming language with many very loyal adherents. For more information check the site at:
- *http://www.smalltalk.org/main.html*