Connecting with Computer Science

Greg Anderson
David Ferro
Robert Hilton

12

chapter

software engineering

Objectives

- Learn how software engineering is used to create applications
- Learn some of the different software engineering process models
- Understand what a design document is and how it should be used during software development
- Be introduced to the proper steps in formulating a design document used in creating an application

Objectives (continued)

- Learn how Unified Modeling Language (UML) diagrams can be used as a blueprint in creating an application
- See some of the pitfalls in developing software, and learn how to avoid them
- Gain an understanding of how teams are used in application development

What is Software Engineering?

- The process of producing software applications
  - Source code and data
  - UML diagrams and screen prototypes
  - Requirements and reports
  - Future development issues
- The end user is the driving force behind software development
  - Determines the required functionality of the program

Software Development Life Cycle

- Describes the life of the application including the development, testing, installation, and maintenance of the program
- Elements of the SDLC
  - Project feasibility
  - Software specification
  - Software design and implementation
  - Software validation
  - Software evolution

Software Development Life Cycle (continued)

- Models of the software development process
  - Waterfall
    - The output from each development phase is used as the input to the next phase
  - Build and fix
    - The developer writes a program and continues to modify it until the system is functional
  - Rapid prototyping
    - Tools allow end users to work with user interface prototypes before building the final product

Software Development Life Cycle (continued)

- Incremental
  - Development with a series of software releases
- Spiral
  - Cycles through the waterfall approach until all functionality has been completed
- Extreme programming (XP)
  - Places emphasis on teamwork and feedback
- Agile Software Development
  - Focuses on customer satisfaction through continual delivery and modifications of software products

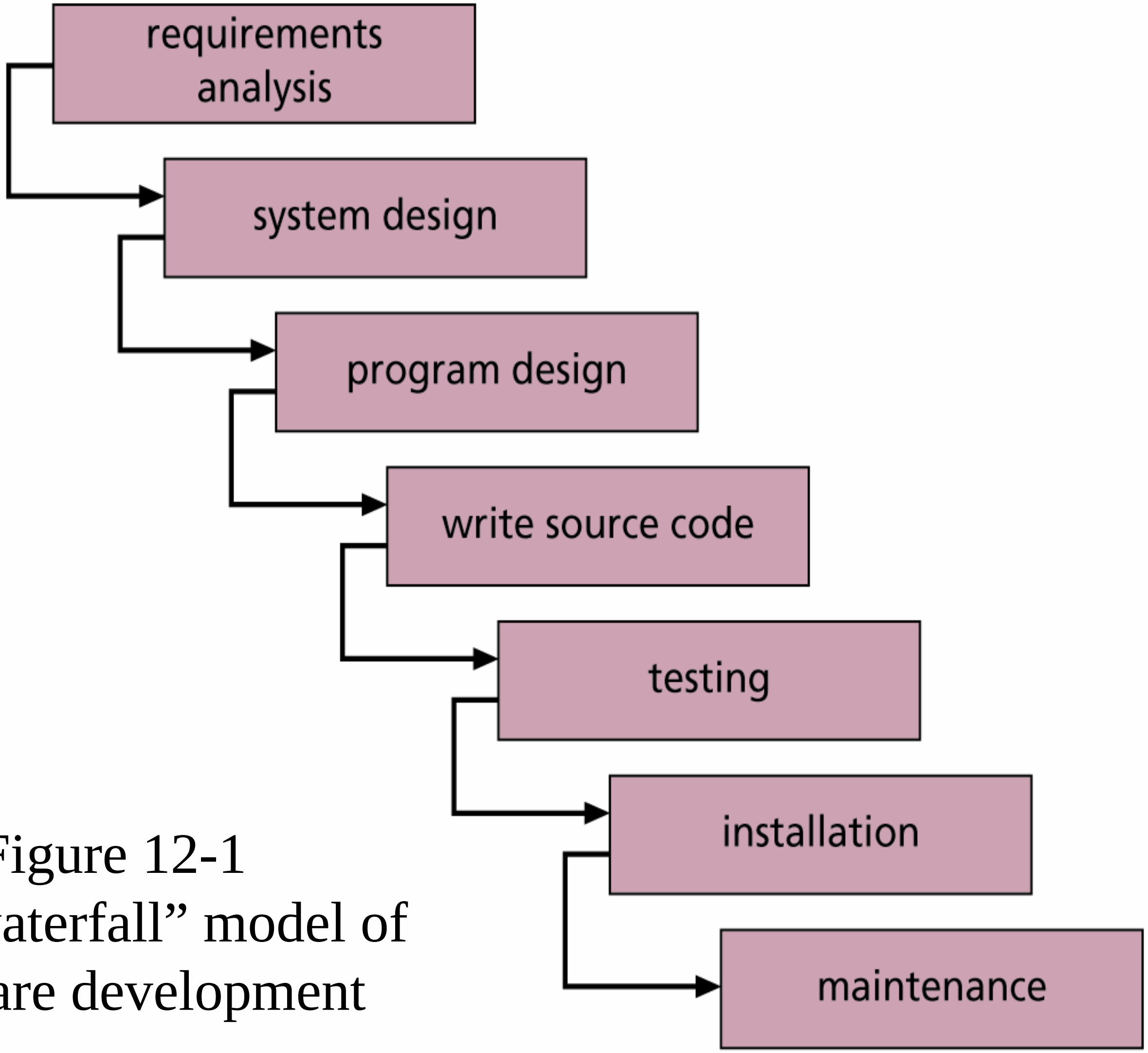


Figure 12-1
The “waterfall” model of software development

Creating the Design Document

- A design document details all of the design issues for an application
  - Issues include screen layouts, colors, reports, security, paths for files, online help, user documentation, future plans, and more
- Acts like a blueprint for the system
- Based upon good communication with the end user in determining requirements

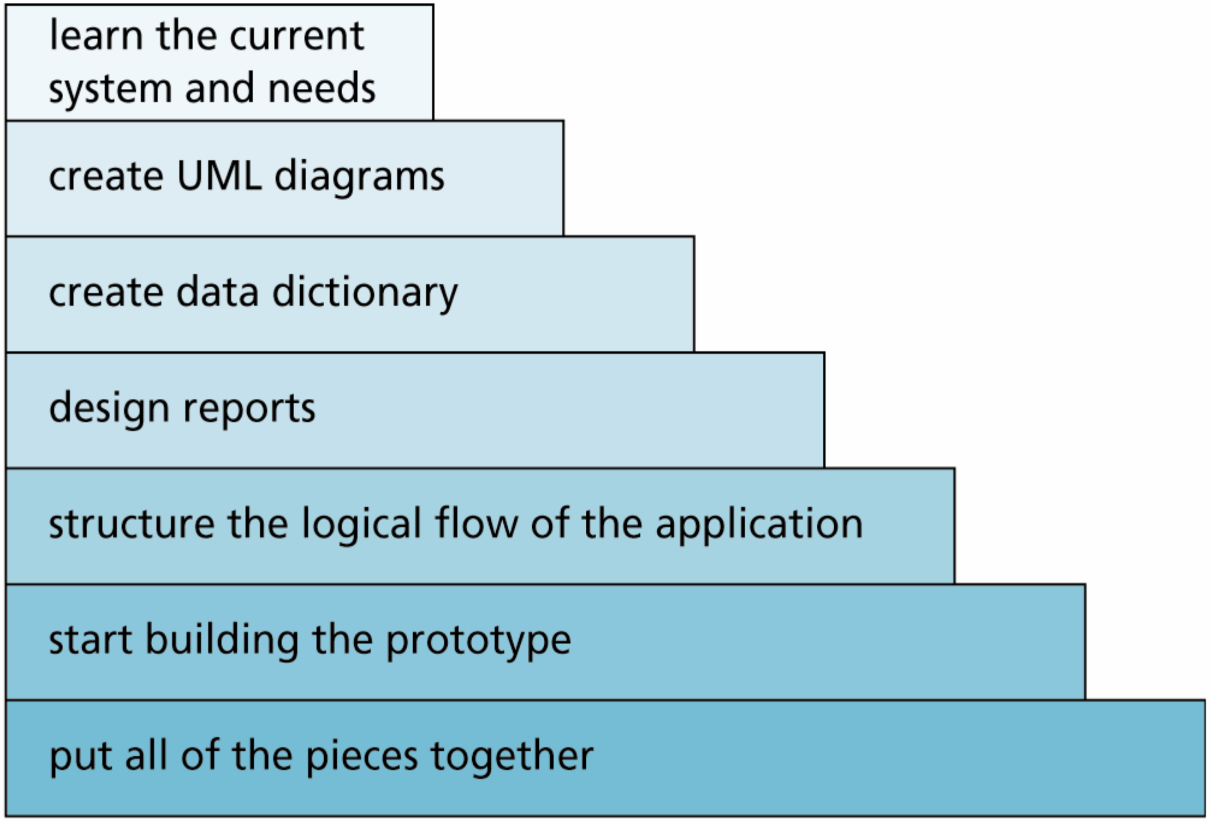
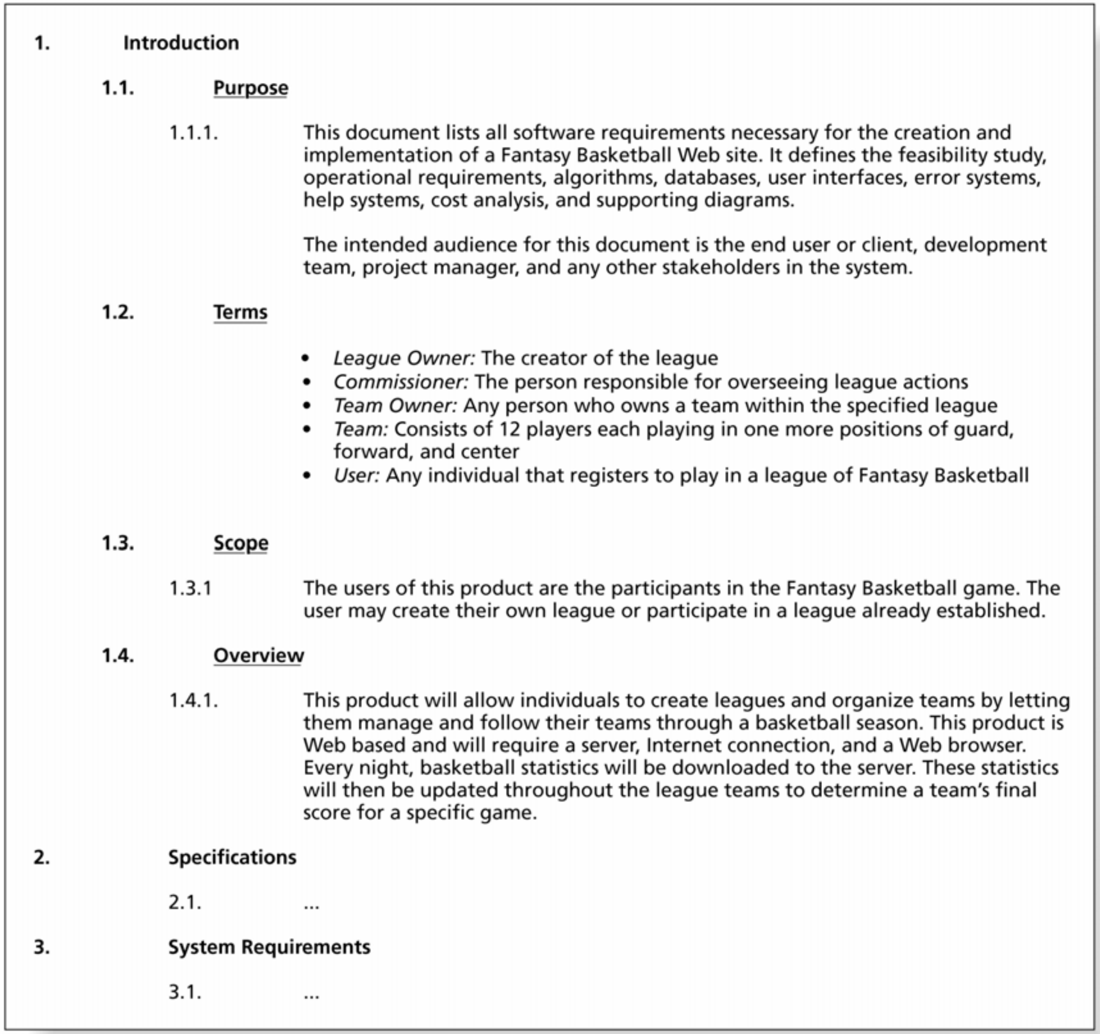

learn the current system and needs
create UML diagrams
create data dictionary
design reports
structure the logical flow of the application
start building the prototype
put all of the pieces together

Figure 12-2
Process of creating a design document

Step 1: Learn the Current System and Needs

- Case Study: a media inventory for a music store
- Initial task is to establish the end user’s (owner of the music store) needs and goals
  - Ask client to give you samples of desired reports
- Document information client gives you
  - Dig for more information if necessary
- Write the project objectives or introduction, specifications, and requirements



1. **Introduction**

1.1. **Purpose**

1.1.1. This document lists all software requirements necessary for the creation and implementation of a Fantasy Basketball Web site. It defines the feasibility study, operational requirements, algorithms, databases, user interfaces, error systems, help systems, cost analysis, and supporting diagrams.

The intended audience for this document is the end user or client, development team, project manager, and any other stakeholders in the system.

1.2. **Terms**

- *League Owner:* The creator of the league
- *Commissioner:* The person responsible for overseeing league actions
- *Team Owner:* Any person who owns a team within the specified league
- *Team:* Consists of 12 players each playing in one more positions of guard, forward, and center
- *User:* Any individual that registers to play in a league of Fantasy Basketball

1.3. **Scope**

1.3.1 The users of this product are the participants in the Fantasy Basketball game. The user may create their own league or participate in a league already established.

1.4. **Overview**

1.4.1. This product will allow individuals to create leagues and organize teams by letting them manage and follow their teams through a basketball season. This product is Web based and will require a server, Internet connection, and a Web browser. Every night, basketball statistics will be downloaded to the server. These statistics will then be updated throughout the league teams to determine a team's final score for a specific game.

2. **Specifications**

2.1. ...

3. **System Requirements**

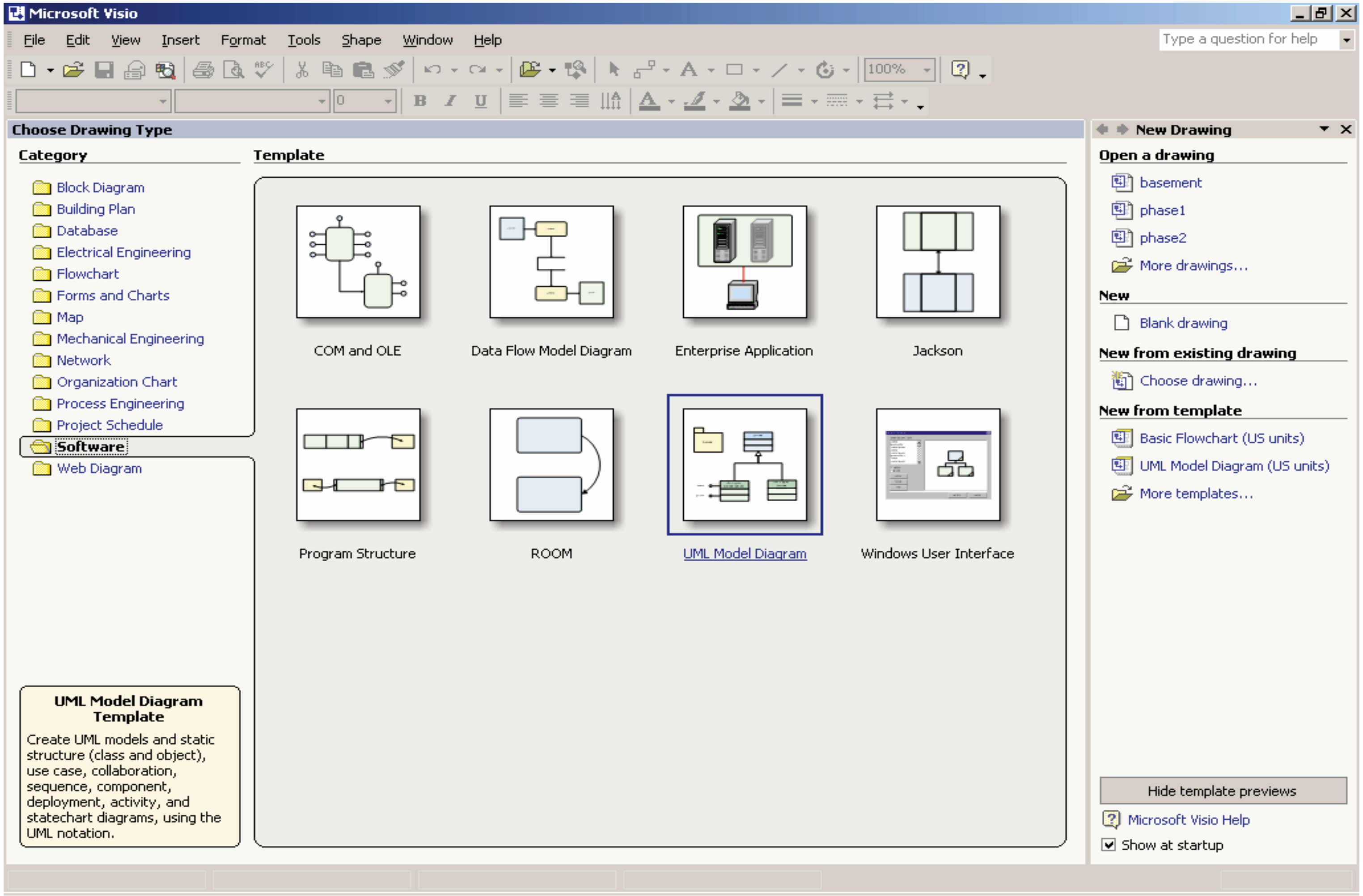
3.1. ...

Figure 12-3

A design document includes the project introduction, specifications, and requirements

Step 2: Create UML Diagrams

- Unified Modeling Process (UML) enables developers to create a visual blueprint of the functionality of the system
  - Provides a way for the client and developer to communicate
  - Visual diagrams are created before any lines of source code are written
- Tools such as Microsoft Visio can be used to create UML diagrams



UML Model Diagram Template

Create UML models and static structure (class and object), use case, collaboration, sequence, component, deployment, activity, and statechart diagrams, using the UML notation.

Figure 12-4
Creating UML diagrams in Microsoft Visio

Some Types of UML Diagrams

- Class
  - Shows how different object classes relate to each other
- Object
  - Gives details of an object created from a class
- Use case
  - Describes a system’s behavior from a user’s standpoint

Types of UML Diagrams (continued)

- State
  - Shows an object’s particular state at any given time
- Sequence
  - Shows how one class communicates with another class by sending messages back and forth
- Activity
  - Shows activities that occur within a use case or within an object’s behavior

Types of UML Diagrams (continued)

- Collaboration
  - Shows how system elements work together to accomplish the system’s objectives
- Component
  - Shows how system components relate to each other
- Deployment
  - Shows physical architecture of a computer-based system

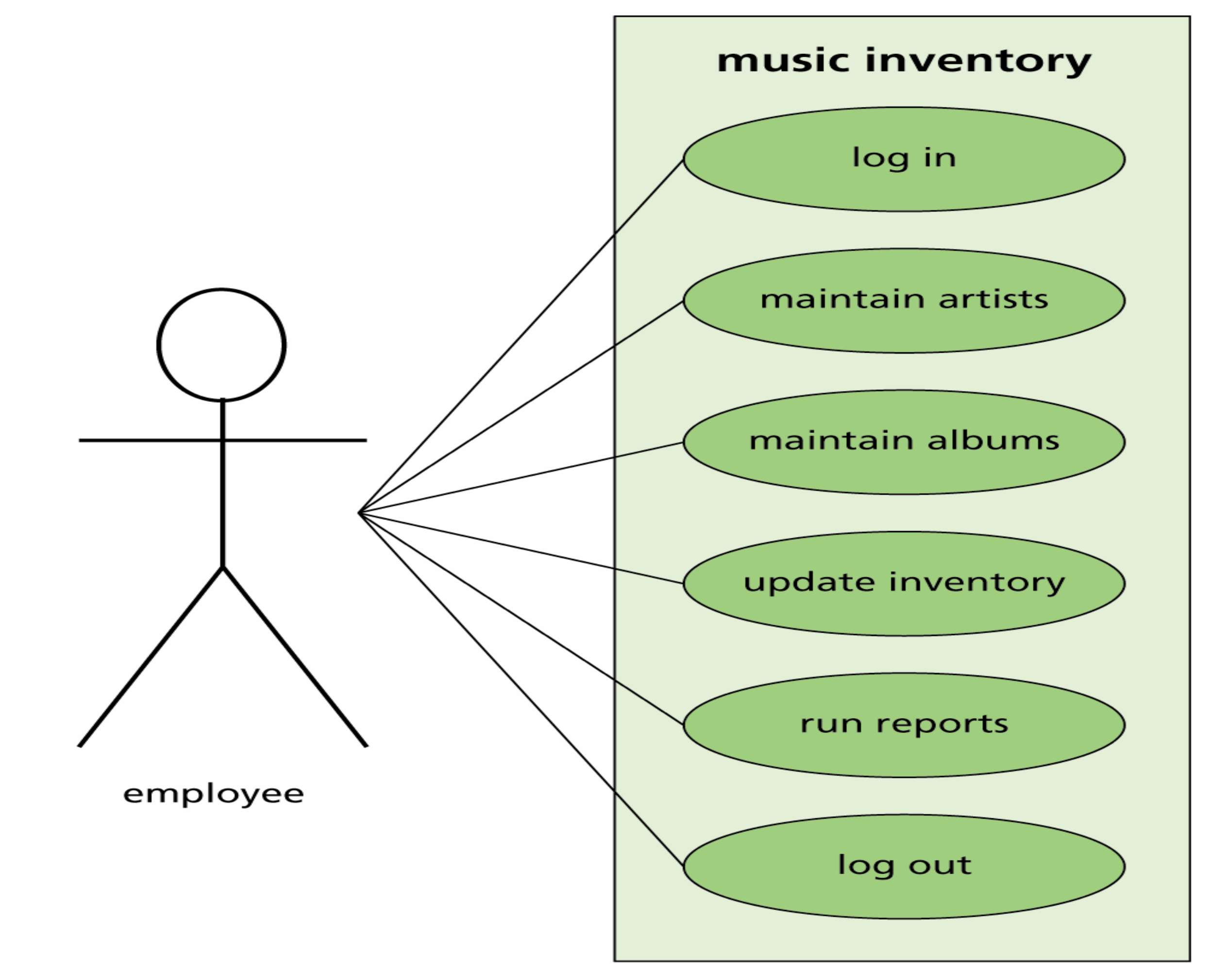


Figure 12-5
Use case diagram for music inventory application

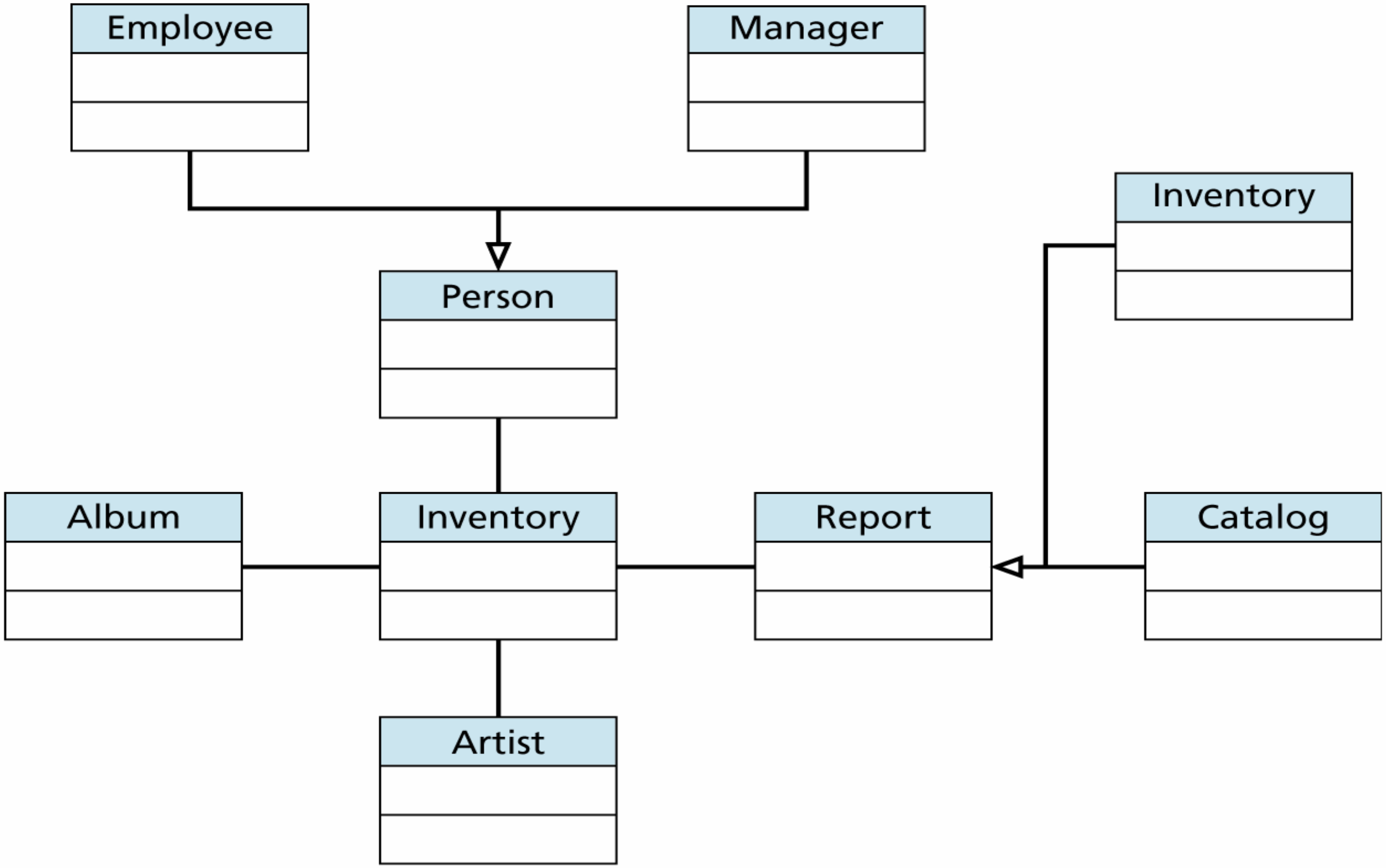


Figure 12-6
Class diagram for music inventory application

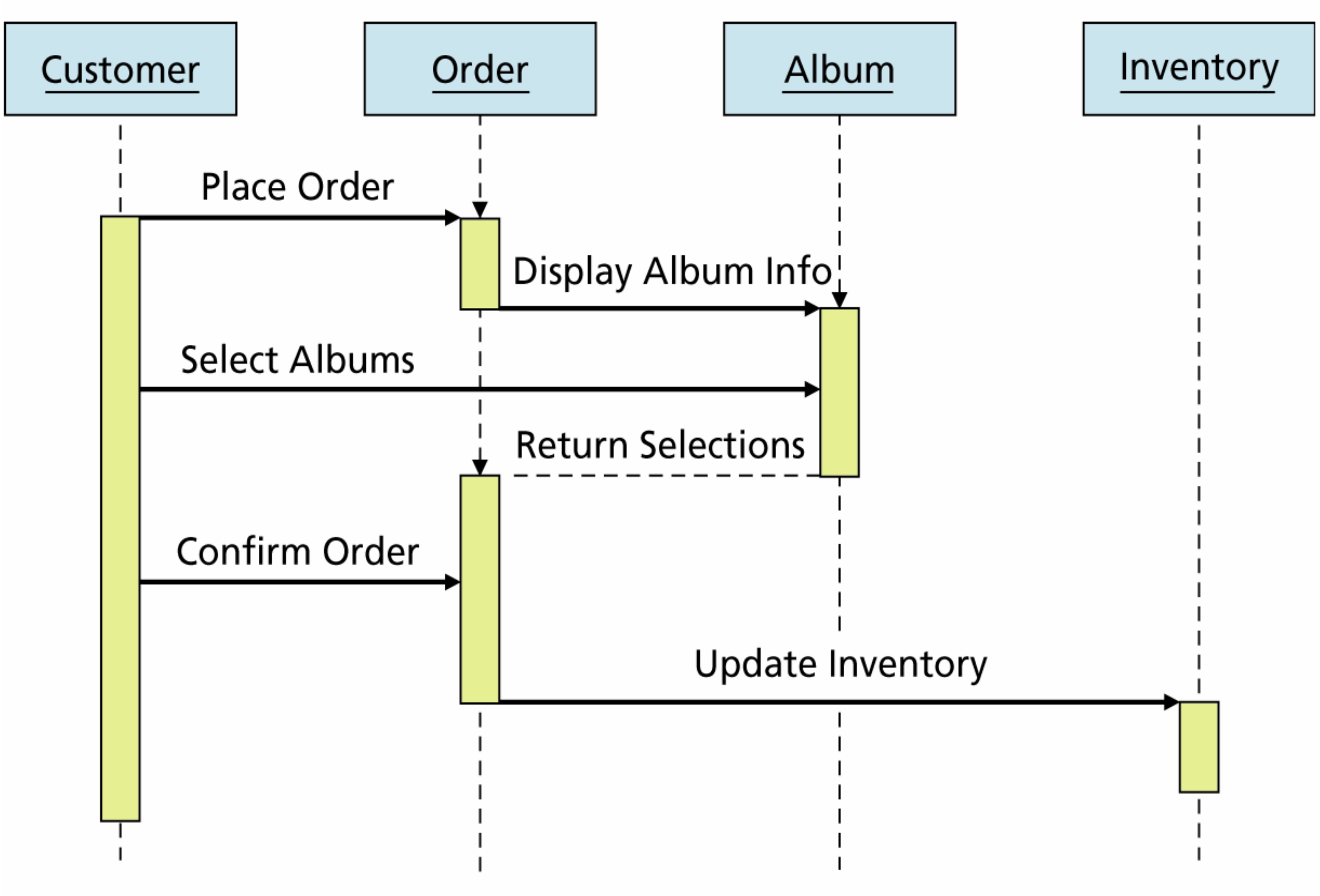
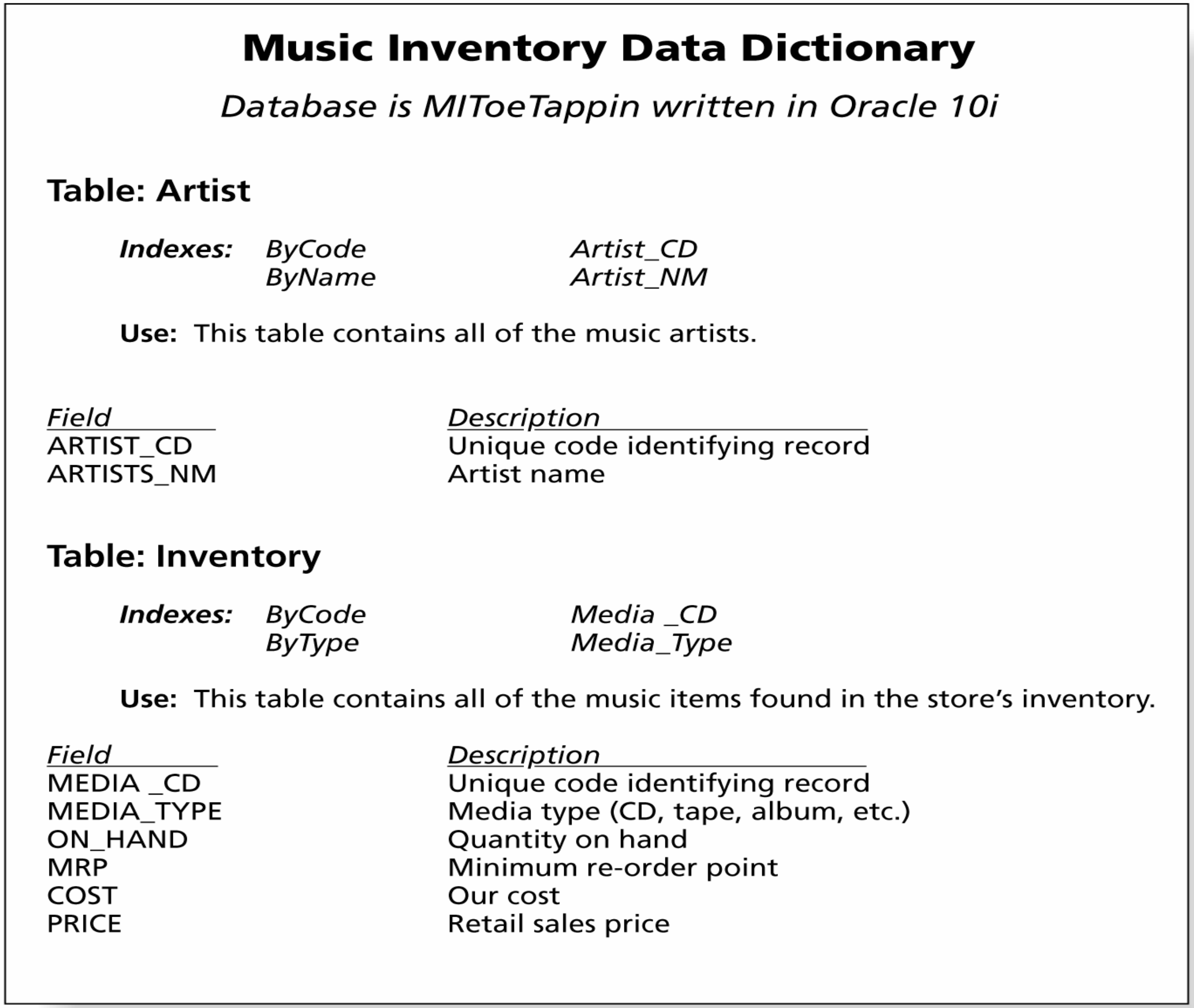


Figure 12-7
Sequence diagram for music inventory application

Step 3: Create the Data Dictionary

- If a database is needed, create a data dictionary
  - Document describing the type of data being utilized within the program
    - Shows table definitions, indexes, and other data relationships
- Use information from end users to summarize the current system and organize a brief plan for the new application
- Review end-user reports to find tables and elements



Music Inventory Data Dictionary

Database is MIToeTappin written in Oracle 10i

Table: Artist

Indexes:	*ByCode*	*Artist_CD*
	ByName	*Artist_NM*

Use: This table contains all of the music artists.

Field	*Description*
ARTIST_CD	Unique code identifying record
ARTISTS_NM	Artist name

Table: Inventory

Indexes:	*ByCode*	*Media _CD*
	ByType	*Media_Type*

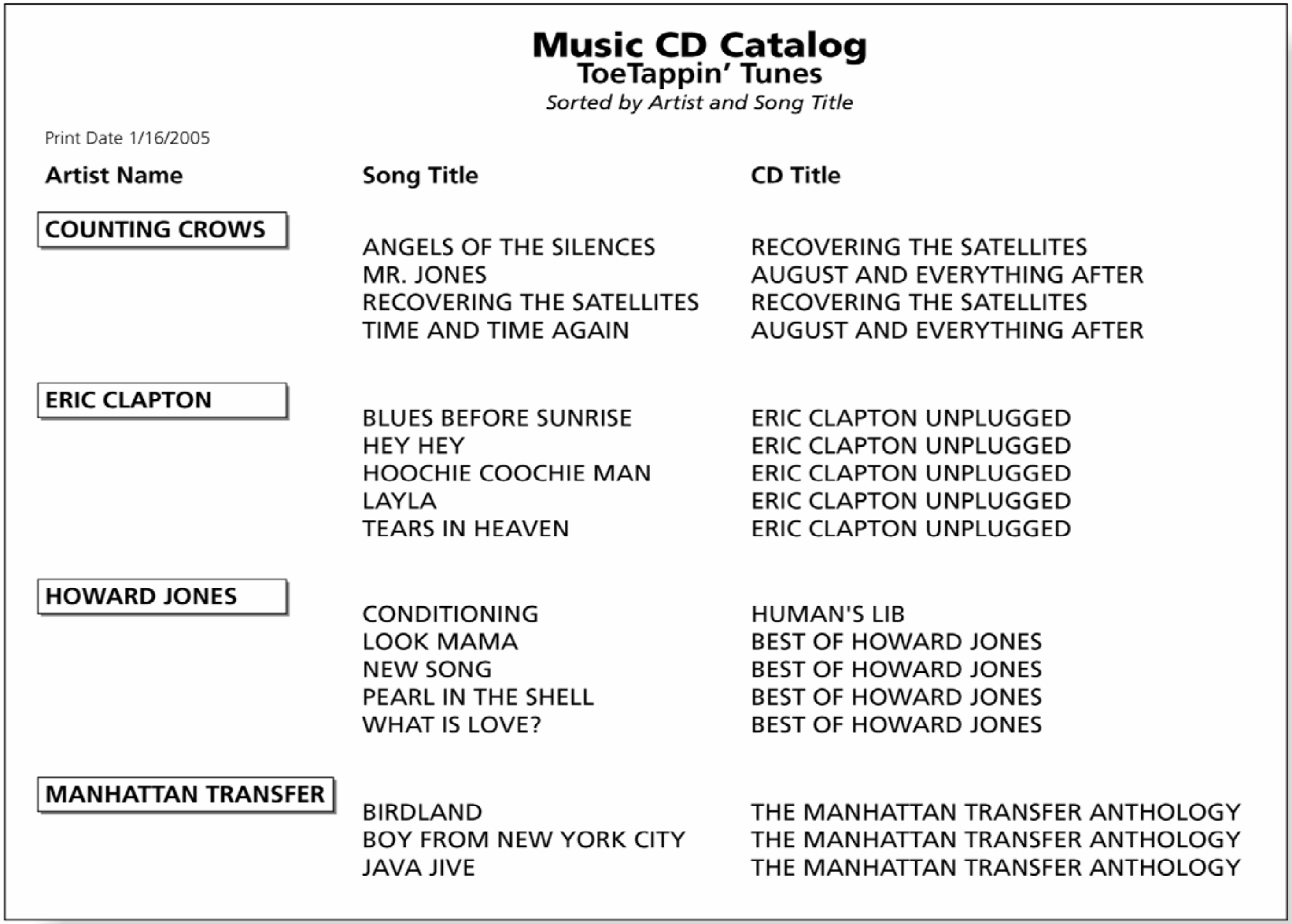
Use: This table contains all of the music items found in the store's inventory.

Field	*Description*
MEDIA _CD	Unique code identifying record
MEDIA_TYPE	Media type (CD, tape, album, etc.)
ON_HAND	Quantity on hand
MRP	Minimum re-order point
COST	Our cost
PRICE	Retail sales price

Figure 12-8
Creating a data dictionary

Step 4: Design Reports

- Meet with users and let them help you design the reports
  - Review data dictionary
  - Use an Integrated Development Environment (IDE)
    - Use the reporting tools in the IDE to interactively create report prototypes with the users



Music CD Catalog
ToeTappin' Tunes
Sorted by Artist and Song Title

Print Date 1/16/2005

Artist Name	Song Title	CD Title
COUNTING CROWS	ANGELS OF THE SILENCES	RECOVERING THE SATELLITES
	MR. JONES	AUGUST AND EVERYTHING AFTER
	RECOVERING THE SATELLITES	RECOVERING THE SATELLITES
	TIME AND TIME AGAIN	AUGUST AND EVERYTHING AFTER
ERIC CLAPTON	BLUES BEFORE SUNRISE	ERIC CLAPTON UNPLUGGED
	HEY HEY	ERIC CLAPTON UNPLUGGED
	HOOCHIE COOCHIE MAN	ERIC CLAPTON UNPLUGGED
	LAYLA	ERIC CLAPTON UNPLUGGED
	TEARS IN HEAVEN	ERIC CLAPTON UNPLUGGED
HOWARD JONES	CONDITIONING	HUMAN'S LIB
	LOOK MAMA	BEST OF HOWARD JONES
	NEW SONG	BEST OF HOWARD JONES
	PEARL IN THE SHELL	BEST OF HOWARD JONES
	WHAT IS LOVE?	BEST OF HOWARD JONES
MANHATTAN TRANSFER	BIRDLAND	THE MANHATTAN TRANSFER ANTHOLOGY
	BOY FROM NEW YORK CITY	THE MANHATTAN TRANSFER ANTHOLOGY
	JAVA JIVE	THE MANHATTAN TRANSFER ANTHOLOGY

Figure 12-9

Example of report created with report generator

Step 5: Structuring the Logical Flow of the Application

- Create logical flow of application before you begin writing source code with
  - Flowcharts
    - Combination of symbols and text that provide a visual description of a process
  - Pseudocode
- Detail the main functionality of the system and the relationship of the different tasks to be completed

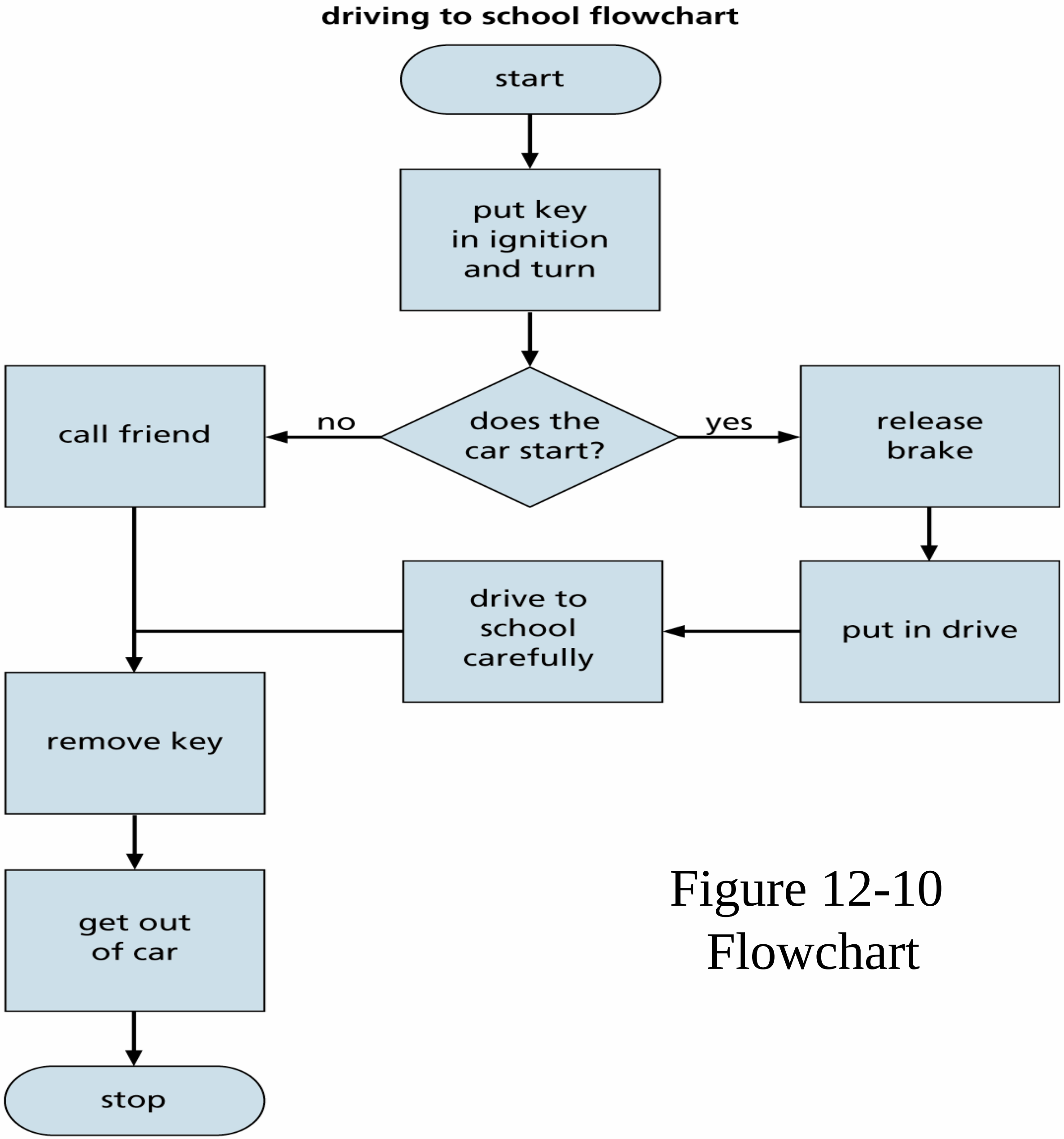


Figure 12-10
Flowchart

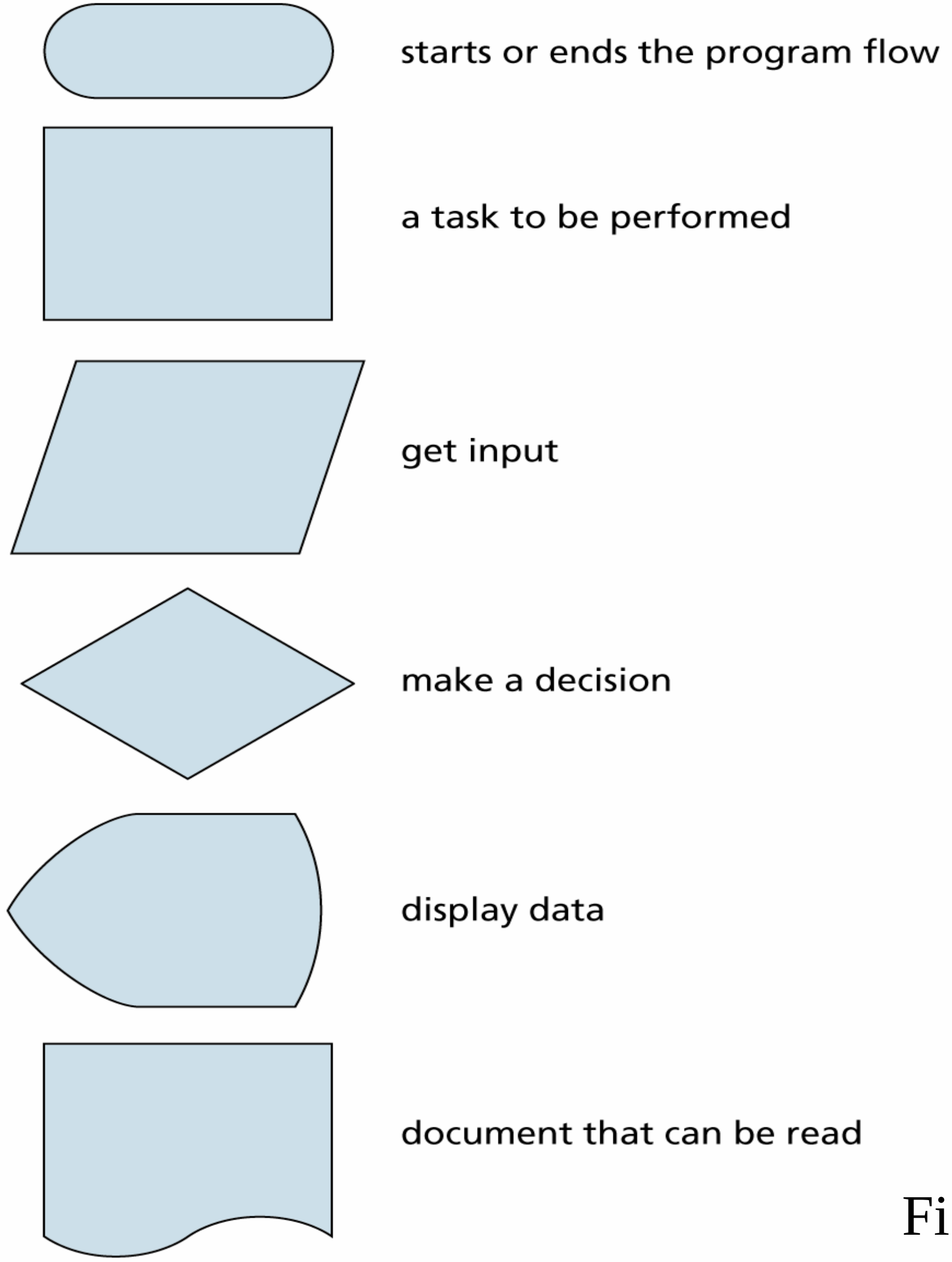


Figure 12-11
Flowchart symbols

Step 6: Start Building the Prototype

- Prototype: typical example that gives end users a good idea of what they will see when their application is completed
  - Ask end users as many questions as possible to understand main functionality of the program
  - Do not write source code until project specifications are approved!
  - Let the user help you design the screens
    - Opening screen should reflect the purpose of the client and main function of the program

Step 7: Putting All of the Pieces Together

- Take all of the gathered information and create the design document
  - Be realistic in defining project completion dates, time lines, and price estimates
  - Make user sign the design document, thus agreeing to the deadline defined by the scope of the project
  - Create addendums when modifications to the document are needed

Step 7: Putting All of the Pieces Together (continued)

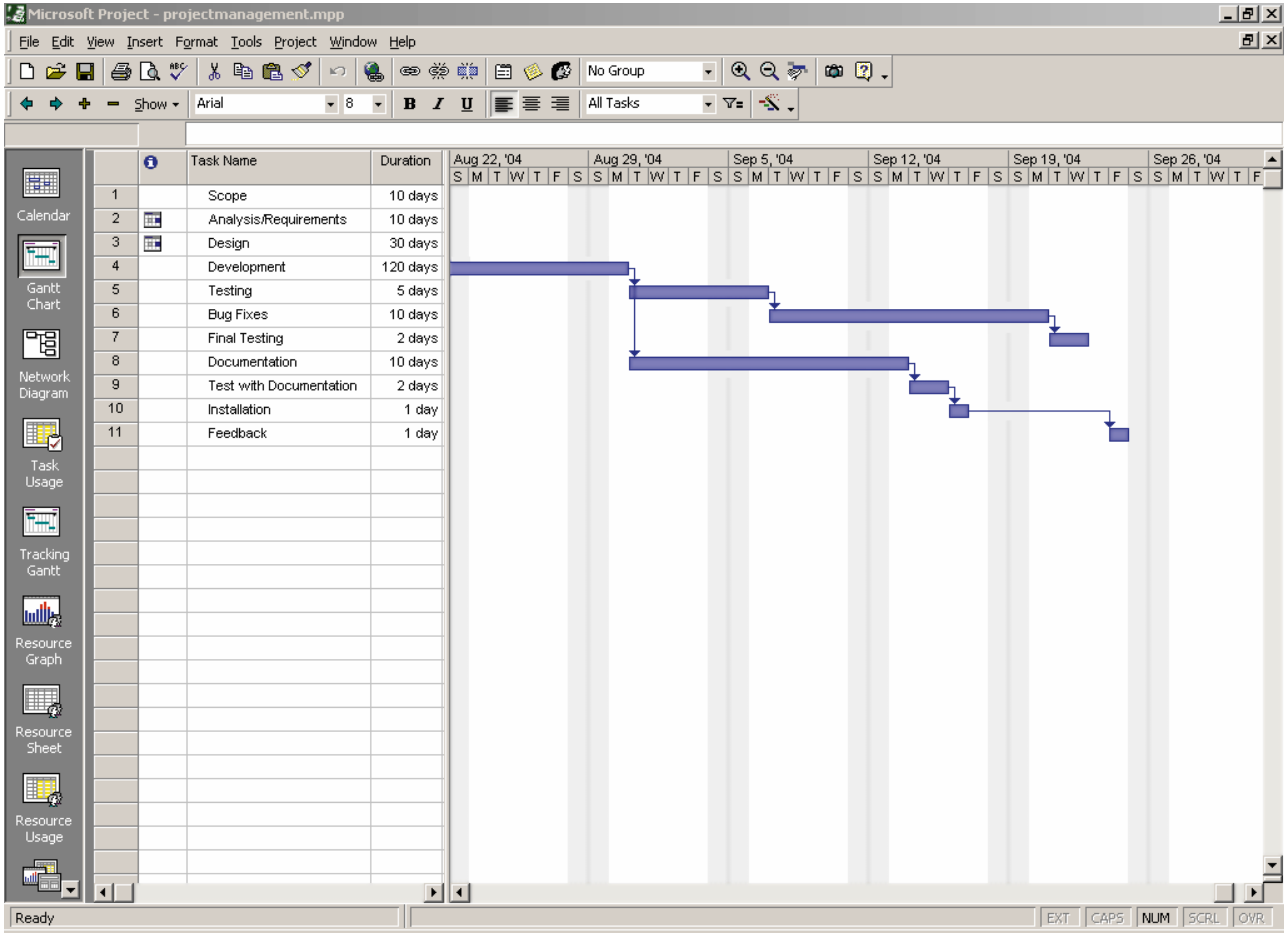
- Include the following items in the design document
  - Header page, objective, definition of terms
  - Project feasibility study, specifications, requirements, and cost analysis
  - Data dictionary and all visual diagrams
  - Copies of screens, and reports
  - Testing and user feedback plans
  - Notes from meetings

Avoiding the Pitfalls

- User-phobia
  - Fears that including end user in design process will result in failure
  - Solution: Keep communication open
- Too much work
  - The “heap on the work” syndrome
  - Solution: Be assertive and honest with the manager
- Scope creep
  - Making continual changes and extensions
  - Solution: Use a phased approach to development

The Project Development Team

- Project manager
  - Responsible for choosing the right players for the right positions
  - Determines risk involved with project, cost, and the scheduling of tasks
  - Pulls together the design document
- Creator of the database (database administrator)
  - Creates and maintains database structure



	Task Name	Duration
1	Scope	10 days
2	Analysis/Requirements	10 days
3	Design	30 days
4	Development	120 days
5	Testing	5 days
6	Bug Fixes	10 days
7	Final Testing	2 days
8	Documentation	10 days
9	Test with Documentation	2 days
10	Installation	1 day
11	Feedback	1 day

Figure 12-12
A project management tool helps a manager keep track of the status of the project

The Project Development Team (continued)

- Developers (programmers)
  - Responsible for writing source code to meet functional requirements of the end user
- Client (end user)
  - Driving force behind project
- Tester
  - Responsible for making sure the program functions correctly and meets all functional requirements in the design document
    - Test all possible situations and keep a log of errors

The Project Development Team (continued)

- Customer relations representative
  - Interface between the testers, developers, and end users during product creation and early release cycles
- Generator of the application CD
  - Interface with developers to ensure that all of the necessary files are included on the disks
- Installer of the application disks
  - Installs disk on the end user’s machine and gives the user a guided tour of the application

One Last Thought

- Good design results in good programs
  - Skipping steps will result in poor performance, unmet client needs, or a project that runs over budget and over schedule
- The project manager’s primary responsibilities are to build a team that can work well together, and to keep the project on schedule and within budget
- Include a thorough testing cycle

Summary

- Software engineering involves many different steps that can be followed in order to create an application that meets an end user’s needs
- The process of building the application is accomplished by implementing a software development life cycle (SDLC) process
- Each SDLC provides a different way of detailing the steps used in creating a software product

Summary (continued)

- A design document acts as a blueprint for software development
- Several steps when creating a design document
  - Researching the needs of the end user, communication, logical design of screens, reports, and data structures, etc.
- UML enables developers and end users to visually design the functionality of the system

Summary (continued)

- Several types of UML diagrams, each for a particular purpose or a part of a system
- Using the reports and the data dictionary can help a developer assess any design “holes”
- Software development is often a team effort
  - Team members include a project manager, database administrator, developer/programmer, client/end users, a tester, and sometimes customer relations

Summary (continued)

- Once the application has been developed, installation disks need to be generated
- After the system is installed on the client’s system, spend some time training initial end users, who can in turn train other end users