

Object Oriented Design

Judene Pretti

tjbailey@cs.uwaterloo.ca

School of Computer Science

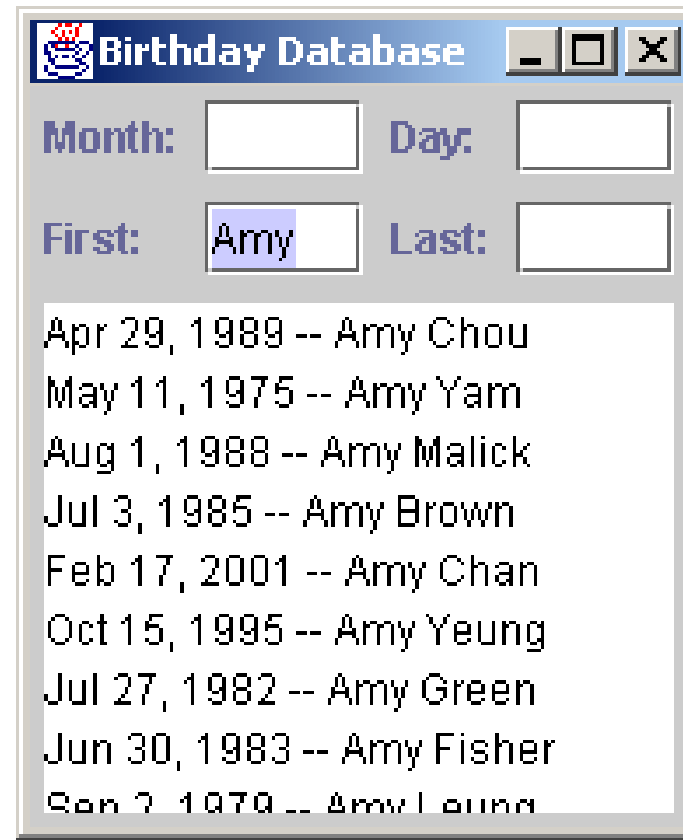
University of Waterloo

Table of Contents

- 1 Project Life Cycle
- 2 Discovering Classes
 - 2.1 The Birthday Database Specification
 - 2.2 Discovering Classes using Nouns
 - 2.2.1 Identify Nouns and Noun Phrases
 - 2.2.2 Guidelines for identifying classes:
 - 2.2.3 Applying the Guidelines
- 3 Classes Responsibilities and Collaborator
 - 3.1 Statements of Purpose
 - 3.2 Identifying Attributes
- 4 Behavioural Responsibilities and Collaborator
 - 4.1 Finding Verbs
 - 4.2 Apply Guidelines
 - 4.3 Assigning to Classes
 - 4.4 Collaborations
- 5 Use Cases
 - 5.1 About Use Cases
 - 5.2 Identify Use Cases
 - 5.3 Conduct a Walk Through
 - 5.4 Summary
- 6 UML Diagram
- 7 CRC vs. Class Diagrams
- 8 The Project
- 9 References

Birthday Reminder Specification

Write a personal birthday reminder program. The names and birth dates of people will be kept in a file named “bdays.txt”. The user should be able to type in a month and day to display all the people born on that day of the year. Alternatively, the user should be able to enter a first name, last name or both to find all the people with matching names.



(Noun = “a word used to identify persons, places or things”)

Write a personal *birthday reminder program*. The *names* and *birth dates of people* will be kept in a *file named “bdays.txt”*. The *user* should be able to type in a *month* and *day* to display all the *people born on that day of the year*.

Alternatively, the *user* should be able to enter a *first name*, *last name* or *both* to find all the *people with matching names*.

Nouns and Noun Phrases

- birthday reminder program
- names of people
- birth dates of people
- file named “bdays.txt”
- user
- month
- day
- people born on that day of the year
- first name
- last name
- people with matching names

Getting Started

- Identify nouns and noun phrases.
- Group/combine similar terms and change plural nouns to singular. Choose the most meaningful noun to represent the entire group.

Identify Potential Classes

Choose	Avoid
<ul style="list-style-type: none">• nouns representing physical objects.• nouns representing cohesive conceptual entities (e.g. a bank account).	<ul style="list-style-type: none">• nouns that are extraneous to the problem or obvious nonsense.• primitive types (numbers, booleans) and strings—they might be attributes of a class, but not a class.• things the program interacts with but are not part of the program (e.g. people, other computer systems). An interface might be needed.• the use of adjectives (e.g. the <i>red</i> wagon). They can make the same thing sound different or different things sound the same.

Working with List of Noun/Noun Phrases

- birthday reminder program
- names of people
- birth dates of people
- file named “bdays.txt”
- user
- month
- day
- people born on that day of the year
- first name
- last name
- people with matching names

Classes: Find the classes by looking at the nouns and noun phrases in the program specification.

Responsibilities: Responsibilities include the knowledge an object maintains (attributes) and the actions it can perform (services).

Collaborations: When an object is asked to do something, it might be able to do it all by itself. If not, it must collaborate with other objects to get the job done.

Write the purpose of the class on the back of each card.

These are recorded at a level of detail that will fit on a note card.

<Class Name>	
Responsibilities	Collaborations

CRC Card

Person:	
Responsibilities	Collaborators

CRC Card

Date:	
Responsibilities	Collaborators

CRC Card

Interface:	
Responsibilities	Collaborators

CRC Card

PersonList:	
Responsibilities	Collaborators

Write a personal birthday reminder program. The names and birth dates of people *will be kept in a file* named “bdays.txt”. The user should be able to *type in a month and day* to *display all the people born on that day of the year*.

Alternatively, the user should be able to *enter a first name, last name or both* to *find all the people with matching names*.

- write a personal birthday reminder program
- will be kept in a file
- type in a month and day
- display all the people born on that day of the year
- enter a first name, last name or both
- find all the people with matching names

Guidelines

- Eliminate irrelevant phrases.
- Restate each phrase to use an active verb which clearly represents an action the system must perform. If it can't be cast as something the system does, eliminate it.
- Examine the class names and purpose statements for responsibilities not exposed by verbs alone.

Assigning Responsibilities

Person	
Responsibilities	Collaborators
read from file	

Date	
Responsibilities	Collaborators

Interface	
Responsibilities	Collaborators
Accept month, day	
Accept names	
Display persons, given name	
Display persons, given birthdate.	

PersonList	
Responsibilities	Collaborators
find, given name	
find, given date	
read file	

Collaborations

Person	
Responsibilities	Collaborators
read from file	

Date	
Responsibilities	Collaborators

Interface	
Responsibilities	Collaborators
Accept month, day	
Accept names	
Display persons, given name	PersonList
Display persons, given birthdate.	PersonList

PersonList	
Responsibilities	Collaborators
find, given name	Person
find, given date	Person
read file	

Use Cases:

- capture some user-visible function.
- achieve a discrete goal for the user.
- may be “large” or “small”.

Examples:

If the application were a word processor, some example use cases could include:

- make some text bold
- delete the previous character
- print the document
- prepare an index
- insert a picture

A large application may have hundreds of use cases.

Use Cases for Birthday Reminder Program

- Start the program.
- Display people matching a given birth date.
- Display people matching given name(s).
- Quit the program.

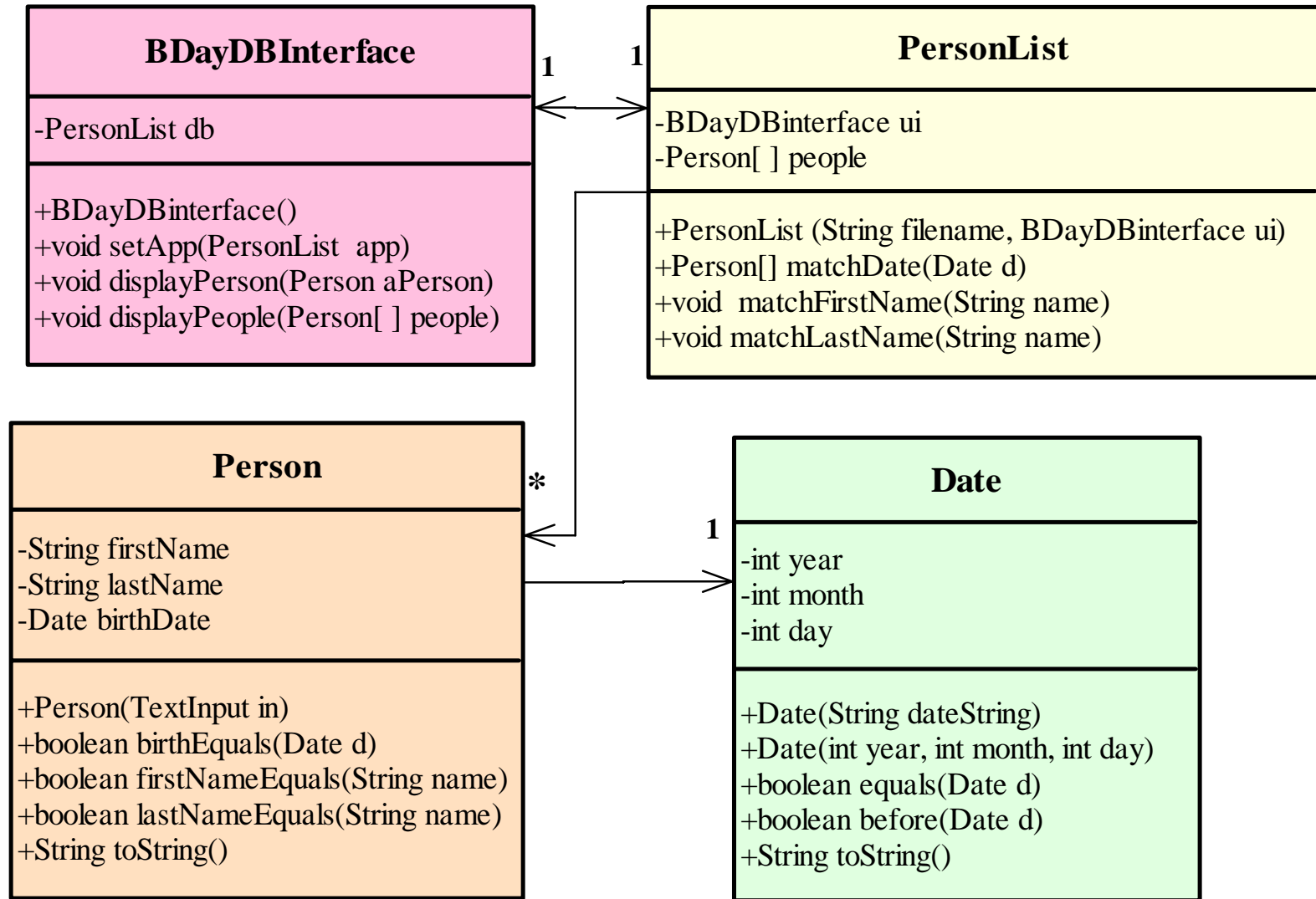
A **walk through** starts with a *use case* and follows the execution through the application.

At each stage it asks the questions of

- which object has responsibility for this?
- which other objects does it collaborate with to carry out the responsibility?

Suggestion: It's great if each object/class is adopted by a team member. During the walk through it's that person's responsibility to explain how the object's responsibility will be fulfilled. If new responsibilities are identified, write them on the card. Also write down the collaborations.

UML Diagram for Birthday Reminder



Ideas We Use

- Basic requirements part and an enhancements part
 - Usually approximately a 70% - 30% split
 - Provide a list of enhancement to choose from with associated values (5%, 10%, 15%)
 - Provide the opportunity for students to come up with other ideas they have for enhancements
- Show & Tell Session
 - Provide some preparation and suggestions on presentation skills
- Interview Marking
 - Opportunity for students to demonstrate their program for the marking
 - Potential for marker to ask questions about challenges faced
- Providing a simple Graphical User Interface
 - Provided GUI works with the basic requirements program

Potential Pitfalls

- ambiguities in project specs (if not intended)
- errors in provided code
- one partner taking over or not contributing
- students adopting a completely different design
- size of project will be a problem for students who procrastinate

Advice to avoid potential pitfalls

- give yourself lots of time to prepare the specs and provided code
- spend some time giving advice on how to work with others; have them consider pair-programming approach (talk about this **very** briefly)
- set milestones for students (mini-deadlines) where certain parts of the project need to be submitted; examples would include
 - Milestone 1: submit all the stubs for the methods in the project
 - Milestone 2: submit unit test cases for specific classes (we often encourage students to set up main methods in each of their classes with tests before they write the code)
 - Milestone 3: submit specific completed classes
 - Milestone 4: submit the basic requirements

Why Large Projects are Important

- Give students an opportunity to work in teams.
- Helps them develop time management skills
- Consolidate the many techniques they've learned in a single context.
- Give them experience in learning new techniques using resources on the web or in books.
- Give them a chance to demonstrate creativity
- Excellent source for practising their testing abilities
- Gives them a sense of accomplishment (something “real” at the end)

References

- UML (Unified Modeling Language)
“*UML Distilled: Applying the Standard Object Modeling Language*” by Martin Fowler with Kendall Scott (Addison Wesley Longman, 1997)
- CRC (Classes, Responsibilities, Collaborators) Cards
“*Designing Object-Oriented Software*” by Rebecca Wirfs-Brock, et. al
c2.com/doc/oopsla89/paper.html
- Project Examples and Administrative Information
www.student.cs.uwaterloo.ca/~cs131/Current/Project
www.student.cs.uwaterloo.ca/~cs133/Current/Project

