

# Assignment Three: Object-Oriented Software Design (Group Work)

Robert S. Laramee and C. Whyley

December 20, 2011

## Contents

1	Problem Statement and Overview	1
2	About the Client and the Assignments (Functional Specification)	2
2.1	The Client's Functional Specification	2
2.1.1	Types of Views:	2
2.1.2	Types of Events	2
2.1.3	Digital Address Book	3
3	Assignment 3 Tasks	3
3.1	Assignment 3 Deadlines: Electronic and Printed	3
3.2	Object-Oriented Design Report for Assignment 3	4
3.2.1	Candidate Classes and Responsibilities:	4
3.2.2	Class Hierarchy Diagrams and Descriptions:	4
3.2.3	Sub-systems Diagrams and Descriptions:	4
3.2.4	Data File Format Description:	4
3.2.5	Minutes Protocol	4
3.2.6	Software Application Design Report	4
3.2.7	Group Report	5
3.2.8	Digital Copies of Files	5
4	Learning Outcomes and Transferable Skills	5

## 1 Problem Statement and Overview

In this assignment, each group designs an application called a Digital Organizer. The digital organizer has the same basic functionality of a digital calendar and digital address book applications. A detailed list of the requirements specifications is given in Section 2 (Functional Specification).

The software design (A3) and implementation (assignments for next semester) are broken down into multiple phases:

1. **Assignment 3 (this semester): Design:** Each group proposes an object-oriented design for the *entire* application. The design requirements are described in detail in Section 3.2.
2. **Assignment 4 (next semester): Partial Implementation:** Each group implements some of the features described in the feature specification

given in Section 2. The implementation follows the guidelines given next semester.

3. **Assignment 5 (next semester): Hand-Off and Assessment:** Each group then hands their full design and partial implementation to another group. Correspondingly, each group takes over another groups' design and partial implementation. Each group then writes a short report assessing the project work they have taken over with respect to the quality of the design, implementation, and testing. A detailed description of this phase is provided next semester.
4. **Assignment 5 (next semester): Complete Implementation:** Each group implements V2.0 of the system, i.e., the complete set of features described in the functional specification section. Again the implementation follows the guidelines given next semester.
5. **Assignment 6 (next semester): Re-Engineering Implementation:** Assignment 6 is a re-engineering task where each individual implements features which are not yet defined. The full description of assignment 5 will be provided in a future document.

You and your team are building an application for a customer called *Bob*. And Bob is a representative of a company called *Bob Incorporated*.<sup>1</sup> Thus, any questions you have regarding the functionality of your system should be directed to customer Bob. Hint: Wake up Customer Interface Manager.

<sup>1</sup>The Welsh branch of Bob Incorporated is called *Bob Limited*.

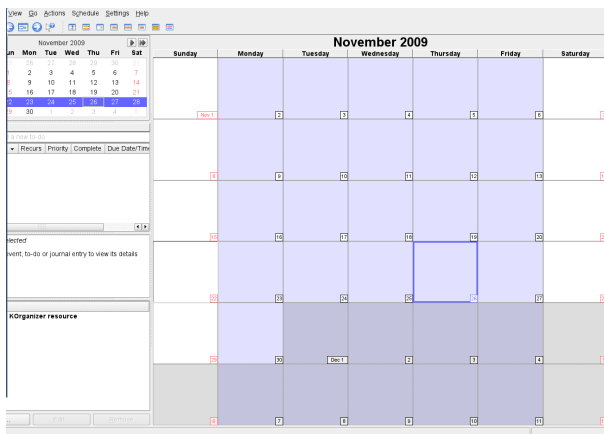


Figure 1: A screen-shot from a sample Digital Organizer application each team is to design.

## 2 About the Client and the Assignments (Functional Specification)

The client requests the development of a digital organizer.

The digital organizer will allow users to store and display information about any events that a user might find important. There will be many types of events, each of which may have different attribute data. In addition to a personal calendar, the digital organizer also features a personal, digital address book.

The overall goal of Assignments 3,4, and 5 will be the delivery of the digital organizer to the client. This will be delivered in multiple phases, a design for Assignment 3, a small prototype (version 1) for Assignment 4 next semester, and a completed prototype (version 2) at the end of Assignment 5 next semester.

These assignments will be completed and assessed in groups. At the end of assignment 4 next semester, the version 1.0 developed by each group will be handed over to a different group for continuing development. The hand-over details will be given a week before the deadline. The contents to be handed-over include all required submissions for the remaining assignments.

In addition to Assignments 3, 4, and 5, there will be an assignment 6 which is to be delivered individually. For assignment 6, the client will request a late change to the software. The details are unknown at this stage,

as the client has no clue about this. Each student will develop his/her own version 3 by making changes to the version 2 developed by their group. A good design for version 1 and version 2 will be helpful to make the development of version 3 easy.

### 2.1 The Client's Functional Specification

Note that this specification is incomplete and sometimes vague as in real-world situations. Each group is required to complete the specification during the design phase of Assignment 3.

#### 2.1.1 Types of Views:

The calendar must have (1) a monthly view, and must have a way for a user to enter a record on an arbitrary date selected from a period. The period must cover at least 1 January 2010 - 31 December 2030. Each group may choose to have a longer or indefinite period.

In addition to the (1) monthly view, the calendar may also have a: (2) yearly view, (3) weekly view, and (4) daily view. All of these views are user options. It is even possible for a calendar to have a continuously moving view according to the clock on the computer. All these views are options. Well developed optional features will be rewarded extra marks. However, the maximum total mark will be constrained by a pre-defined allocation.

#### 2.1.2 Types of Events

The digital calendar should support at least 15 different types of events, including:

1. Social event
2. Birthday
3. Work Deadline (e.g., Assignment, Marking)
4. Meeting
5. Bill Payment
6. Appointment
7. Anniversary
8. Class/Lecture
9. Meal
10. Happy Hour
11. Concert
12. Accident
13. Bank holidays

#### 14. Other (unclassified)

Each group can add more events, such as a Telephone Message, but is advised to limit the total to 25,

Some of the above event types can also be further divided into sub-types (e.g., work general meeting, work management meeting, other meeting).

Each type of event will have (store) a record. The contents of the records may vary from type to type. It is necessary to avoid inappropriate contents in a record. The details of the contents of each type of event is to be determined by each group in the design phase of Assignment 3 using an object-oriented approach to define classes with different contents.

The contents may include but not limited to:

1. Event title
2. Date
3. Starting time
4. Ending time
5. Location
6. How often the event repeats
7. Description
8. Persons involved

Each group can add more contents appropriate for a type, such as a display icon.

#### 2.1.3 Digital Address Book

The digital address book features the following:

1. The ability to (1) add, (2) edit, and (3) delete contacts to/from the address book.
2. The ability to view the complete list of contacts in his address book.
3. The capability of switching easily and seamlessly between calendar views and address book views in the application. In other words, the address book should be a function which is integrated with the digital calendar to form a general digital organizer.

Contact information should include, at a minimum: (1) contact name, (2) address, (3) mobile telephone

number, (4) email address, and optional fields such as (5) home telephone number, (6) fax number, (7) additional email address, and (8) URL. These are the fields that your application supports, not the minimum information that a user must enter in order to create an entry in the address book. The minimum information necessary to create an address book entry is up to you, but it should make sense. For example, an entry without a name does *not* make sense. Also, basic error checking of all input data is necessary. A name should not consist of numbers (as one example).

The digital organizer data is persistent. In other words, it is stored in a file and loaded automatically on application start up.

### 3 Assignment 3 Tasks

For Assignment 3, your team provides a complete **design** for the entire system described in Section 2. It includes a reasonably complete class hierarchy. More detail about the design report is given in Section 3.2.

#### 3.1 Assignment 3 Deadlines: Electronic and Printed

**Number of Credits:** 25% of a 15 credit module

**Recommended Hours:** Approximately 30 hours (per group member)

#### Important Deadlines:

1. **6 December** by 11:00am first copy of minutes of meeting. Minutes are meeting are worth 10% of the assignment.
2. **Design Document (Section 3.2)**
  - (a) **16-December 23 January** by 11:00am: online and printed submissions of design document. The design document is worth 80% of the assignment.
3. **Group Report (Section 3.2.7)**
  - (a) **16-December 23 January** by 11:00am: online and printed submissions of group report. The group report is worth 10% of the assignment.

Printed copies of required coursework documents are placed in the coursework submission drop-box or given to the Departmental Student Secretary in room 206 of Faraday Building.

## 3.2 Object-Oriented Design Report for Assignment 3

The Design Report proposes an object-oriented design for the *entire* application. In other words, your team incorporates the full functional specification (Section 2) into the design. The Application Design Report is no more than 20 pages including text and diagrams and consists of the following sections:

### 3.2.1 Candidate Classes and Responsibilities:

Provide a list of candidate classes and their responsibilities. This list follows the class-card style described in the lectures and by Wirfs-Brock et al. [2, 3]. For each candidate class the team has identified, the following information is provided:

1. **Class Name:** (in bold)
2. Author:
3. SuperClass:
4. SubClasses:
5. Responsibilities: a list of services this class provides
6. Collaborations: a list of classes that this class may communicate with
7. Protocols: a list of methods that belong to this class including fully specified method signatures, i.e., method names, input parameters, and return parameters.
8. Unit Tests: some possible ways the class or object can be tested for functionality and robustness

### 3.2.2 Class Hierarchy Diagrams and Descriptions:

Each class hierarchy identified during your design process is depicted in a diagram. The drawing style is to be modelled after the drawing style used in the lectures and by Wirfs-Brock [2, 3]. Your team is also welcome to use the drawing style of UML [1]. Each diagram is accompanied by a short description that describes the “is-kind-of” relationships used. Make sure that your team provides a justification that backs up its choices. Abstract classes are distinguishable from concrete classes in the diagrams. Your team is to identify three (or more) candidate class or object hierarchies.

### 3.2.3 Sub-systems Diagrams and Descriptions:

Each sub-system identified during your design process is depicted in a diagram. Again, the drawing style is to be modelled after the drawing style used in the lectures (and by Wirfs-Brock [2, 3]) or the drawing style of UML [1]. Each diagram is accompanied by a short description that describes the “is-part-of” relationships used and depictions of classes that work closely together. Again, make sure that your team provides justification for each sub-system, in other words, why you think a certain set of classes or objects belongs together in a sub-system. Your team is to identify two-three (or more) sub-systems.

### 3.2.4 Data File Format Description:

Describe the data file format used to store the data and all of their associated attributes. More information about the data is given in Section 2.

### 3.2.5 Minutes Protocol

A copy of your group minutes for three (or more) meetings held before the assignment deadline. Your minutes files should be called “*GroupNminutesD-monYr.txt*” where *N* is replaced by your group number, *D* is replaced by the calendar day your group met, *Mon* indicates the month that the group met, and *Yr* indicates the year, e.g., *Group1minutes23oct11.txt*. (You will be assessed on this.) Send a *link* to your first minutes to customer Bob as an email to reassure him that you are working hard on his product. A minimum of three minutes of meeting protocol is required and are submitted on behalf of the group by one of its members. *The first minutes has a special interim deadline before the other files.* See the given interim deadline. In addition, a digital copy of each minutes file is placed in a folder called `minutes` that resides in the `cs254groupNa3` folder described above.

### 3.2.6 Software Application Design Report

A copy of your Application Design Report online *and in printed form* is required by the deadlines indicated above. The file should be called “*GroupNdesignReport.pdf*” where *N* is replaced by your group number. PDF format is strongly encouraged. Open Office Document Format (.odt) is also acceptable. Word document format (.doc or .docx) is not acceptable. See the following URL to convert word document format to PDF:

<http://cs.swan.ac.uk/~csbob/teaching/cs124-tutorial/>

Attention: you will be assessed on your conformity to the instructions. Don't forget to write the group member names, student numbers, and managerial roles in the report. In addition, a digital copy of this report is placed on the CS web server in a folder called `design` that resides in the `cs254groupNa3` folder described above.

### 3.2.7 Group Report

A description of what and how each group member contributed to the project and the managerial role allocation is submitted. More detail about the group report content is given in Section 3.2.7. The file is called "*GroupNmemberContributions.pdf*" where *N* is replaced by your group number. A digital copy of the group report is placed on the CS web server in a folder called `groupReport` that resides in the `cs254groupNa3` folder described above.

The group report also tells the reader: (1) where they can download the project files from (the URL) and (2) how to compile the code to produce the application. A printed copy of the group report is also submitted.

### 3.2.8 Digital Copies of Files

Thus, when completing the submission of this Assignment, the Planning and Quality Manager has a directory structure in their `public_html` folder that looks like this:

```
.../public_html/cs254groupNa3/  
    design/  
    groupReport/  
    minutes/
```

Make sure that all of the files and folders are accessible (readable and executable) to anyone who has a link to your teams' `public_html` folder: e.g.,

```
%> chmod -R ugo+rx *
```

If you are worried about another group viewing your submission before the deadline, you can use the command:

```
%> chmod u=r *
```

to change the permissions of your files so that only the current user can access them. Issue the first command above on the submission deadline.

Points will be deducted for those submissions that do not follow the file naming conventions and required file formats.

## 4 Learning Outcomes and Transferable Skills

**Learning Outcomes:** Students will gain an understanding of the principles of software engineering; an understanding of the key HCI concepts in the context of system evaluation and design; the ability to design and evaluate GUIs; an understanding of object-oriented programming concepts, and knowledge of their applications in software design and engineering processes; the ability to build GUIs and skills of event-driven programming; experience and appreciation of group work; skills of project management.

**Transferable Skills:** Problem solving through analysis and abstract reasoning. The ability to read critically, to precis and judge information. Experience and appreciation of team work, time management, project management, and risk assessment. Skills in written communication and documentation. The ability to learn and use computer systems and software packages effectively.

## References

- [1] M. Fowler. UML Distilled: A Brief Guide to the Standard Object Modeling Language. Object Technology Series. Addison-Wesley, third edition, September 2003.
- [2] R. Wirfs-Brock and A. McKean. Object Design: Roles, Responsibilities, and Collaborations. Addison-Wesley, 2003.
- [3] R. Wirfs-Brock, B. Wilkerson, and L. Wiener. Designing Object-Oriented Software. Prentice-Hall, 1990.