ORDRE DES INGÉNIEURS DU QUÉBEC

MAY 2013 SESSION

All documentation allowed

Pocket calculator : authorized models only

Duration of examination : 3 hours

04-Log-A3 Software design

**Context:** The following text describes the main functionalities of the *ReservABus* software system.

The *ReservABus* software is an application that allows managing bus fleets and the associated personnel. The application is targeted towards companies that offer transport services for the tourism industry through charter bus rentals. The application runs locally on standard personal computers running Windows 7 and stores all its data in a database on a remote server. The application is mainly used by transport company personnel who take reservations from clients over the phone. The application is also used by company managers.

The *ReservABus* application offers the following main functionalities to all users. The application allows to:

1. Open a session and identify each user through a unique login and password,
2. Record a reservation for a given number of buses. A reservation includes the date and place of departure, one or several destinations, waiting time at each destination, and date and place of return. When planning a reservation, the client may choose to also reserve tour guides and specify which languages the guides should speak.
3. Assign, for a given reservation, a driver for each bus,
4. Assign, for a given reservation, a tour guide for each bus,
5. Display a list of recorded reservations for a given calendar period. A calendar period is specified as a beginning time and date and an ending time and date,
6. Display a list of available buses for a given calendar period,
7. Display a list of available drivers for a given calendar period,
8. Display a list of available tour guides who can speak a given language for a given calendar period,
9. Print all information related to a given reservation in a form to be handed out to the drivers and tour guides,
10. Record the time and date of departure of buses involved in a given reservation,
11. Record the time, date of return and distance traveled by each bus in a given reservation,
12. Prepare and print an invoice for the client once the travel service for a reservation has been fully provided.

In addition to the above functionality, which is available to all employees, company managers also have access to some administration functions.

Administration functions of the *ReservABus* application allow to:

1- Create, modify parameters and delete user accounts,
2- Manage the list of buses and their capacity,
3- Manage the list of drivers,
4- Manage the list of tour guides and specify for each which languages they can speak.

**Question 1 – Design Process and Use-Cases (20 points)**

a) Modern software development processes are based on the elaboration of use-cases of the software system to be conceived. In the context of use-cases, define the following terms in your own words, and provide an example for each, drawn from the description of the *ReservABus* application:
   i. A use-case (2 points),
   ii. An actor (2 points),
   iii. A precondition (2 points),
   iv. An association between two use-cases decorated with the <<extend>> stereotype (2 points).

b) Draw an UML use-case diagram of the *ReservABus* software (12 points)

**Question 2 – Logical Architecture (20 points)**

Modern software systems are developed in accordance to layered or multi-tier architectural principles.

- Indicate two important advantages of logically decomposing a software into a layered or multi-tier architecture (4 points),
- In the course of logical architectural design, Larman suggests the use of the *Controller* pattern. Explain the role of the *Controller* and state the advantages brought by its use. (2 points),
- As part of a software system designed according to the principles of a layered or multi-tier architecture, some function or method invocations between modules are allowed, while others are forbidden. Describe and explain the rules that dictate which invocations are allowed and which are forbidden (4 points).
- Propose a logical multi-tier architectural decomposition of the *ReservABus* application. Provide your answer in the form of an annotated UML package diagram (10 points).

**Question 3 – Object-Oriented Analysis and Design (30 points)**

a) In the context of an object-oriented design, define in your own words the following terms :
   i. Base class (1 point),
   ii. Interface (1 point),
   iii. Type (1 point).

b) What distinction do you make between the three concepts defined above? (2 points)

c) Restricting yourself only to the functionality and concepts identified in the description of the *ReservABus* system above, propose a UML class diagram of the *ReservABus* application (25 points).

**Question 4 – Design Patterns (30 points)**

During the analysis and design phase of the *ReservABus* application, the following three design problems have been identified, which you are asked to solve using software design patterns:

1. There exists, in the *ReservABus* application, a complex relationship between objects that represent `drivers` and objects that represent `reservations`. We wish to represent relationships between `driver` and `reservation` objects while maintaining a low coupling between these objects.

2. There exist a great number of possible types of `buses` on the market, each with its own set of characteristics, and we wish to provide a mechanism to construct concrete objects that represent `buses` while isolating the *ReservABus* application from knowing the concrete types of `buses` that are manipulated by the application.
3. When a `driver` or a `guide` is assigned to a reservation by a company employee, the *ReservABus* application must automatically emit a notification message to inform the `driver` or `guide` using an email or text message (SMS), providing all relevant information relating to the new reservation. If the reservation is subsequently modified, all personnel assigned to the reservation must be automatically informed of the change.

For each of the three design problems, you are asked to:

a) Identify the appropriate design pattern (from *Gamma et al.*) that should be used to solve the design problem and justify your choice (2 points),
b) Apply the design pattern by providing an annotated UML class diagram, where roles present in the pattern are explicitly associated to the appropriate classes (4 points),
c) Discuss advantages, drawbacks and consequences of using the chosen pattern to solve the design problem (4 points).