Real Object-Oriented Programming

Homework: "Basics of UML with Rhapsody"

Task 1. Design a class diagram for Payroll Information System.

NB! Many solutions exist for this task. However you need to apply principles of UML diagramming and carefully follow the requirements.

You need to start from worker and project classes; constraints for these are listed below:

Constraints for worker classes:

- The types of workers in the company are*: trainees, hourly employees, project managers, sales managers, permanent employees*. The difference between workers is explained below.
- A *trainee* has an explicit *starting* and *ending date* of work in the contract. *Trainee* has a *supervisor*, which is another worker in the company; however a trainee cannot supervise anybody. Project Managers and Sales Managers can supervise trainees also.
- An *hourly employee* is paid salary depending on *total number of hours* the employee worked during the month.
- A *project manager* is paid salary as a *permanent employee*, but he gets also *bonus* (it depends on number of project he supervises). *Project manager* can supervise *multiple projects* at the same time.
- A *sales manager* is paid salary as a *permanent employee*, plus he gets *5% of bonus* money from monthly sales.
- *All workers* have the following attributes:
  - *Office room*
  - *Email*
  - *Office phone*
  - *Job Position*
  - *Department*
- *All workers* have a work contract, which contains data about *position*, *dates* when employer *started work* and when he will finish it (if the date is known already).

Constraints for project classes:

- *All workers*, besides Sales and Project Managers work at least in *one project*.
- *A project* must have only *one supervisor*.
- *Projects* can be *composite* (can consist of a number of smaller projects). However, a project *cannot include* subprojects that already belong to another project.
- *Each project* has a *name, acronym, starting and ending dates* as attributes

Requirements:

- Do use different types of relationships (e.g., aggregation, inheritance, uni-directional and bi-directional associations).
- For all relationships set cardinality (multiplicity) and other necessary parameters.
- Do make attributes of the classes visible before printing out the diagram.
- All attributes must have a correct data type set (e.g., string, date, etc).
- Do follow the naming convention (the rules are listed at the end of the document)

Hints:

- If several classes have common attributes or methods, then make a superclass that has common attributes and methods, and link original classes to this superclass.
- A class can be associated with itself (can have reflexive relationships).
- If class A is a part of class B, use basic aggregation between them (empty diamond shape).
- If an instance of class A has to be related to several instances of class B, use associations with multiplicity (e.g. the requirement "project manager can supervise multiple projects at the same time" tells that multiplicity of the relationship of "project manager" and "project" classes is one-to-many).
- Make several iterations to refine you solution.
- Read IBM Rational UML tutorial (the link is at the end of the document).

Task 2. Design state chart for worm game application.

NB! Many solutions exist for this task.

The task is to design a state chart for well-known game. You should focus on the states of the application, but not the worm. Start from initial state (application is not initialized) and final state (application is destroyed). Define the states between these two. Then connect states with events and transitions (events can come to the game from "outside" also).

Consider, for example, the following events:

- Start game
- Stop game
- Restart game
- Worm died
- Candy eaten
- Timer tick

Are there any other events or transitions? (Time events, signals, self-transitions).

Task 3. Design sequence diagram for Library Information System

NB! Many solutions exist for this task.

There are the following objects: user, application interface, database and barcode scanner interface. Consider the scenario when user comes to borrow a book (he knows book's titles), he logs in to the system, browses the interface, finds the book in the catalogue, brings the book from the shelf and scans the code of the book and his library card, etc.

- Do you need more objects for the system?
- What is the possible flow of messages and method calls between these objects?

Design the complete sequence diagram that describes behaviour of the system in this scenario.

How to return the homework

- The homework is performed ***individually***. Therefore, identical homework solutions noticed cause the whole exercise to be unaccepted.
- ***Type*** your student number, name and surname on title list (first diagram).
- Print the diagrams for tasks 1-3 and ***bring your solution with you*** to the exercise.
- Be ready to comment your solution!

UML naming conventions for class diagram:

1.1 Class names

- Starts with an uppercase letter
- Starts with a noun
- Every word that composes the name starts with an uppercase letter
- Words are run together (e.g., DomainManager)

1.2 Methods and attributes

- Starts with a lowercase letter
- Method's name starts with a verb
- Attribute's name starts with a noun
- Words are run together (e.g., getApplications)
- Every word that composes the name starts with an uppercase letter (except the first one)

1.3 Associations

- Starts with an uppercase letter
- Starts with a verb
- Every word that composes the name starts with an uppercase letter
- The name of an association should not be a concatenation of the names of the messages that the classes can exchange. So an association name like "registers itself, devices, and services to" is not a proper association name.

More Information on UML

If you need extra help on UML, try these:

- [UML resource page] http://www.uml.org
- [IBM Rational UML tutorial] http://www-306.ibm.com/software/rational/uml/
- [Borland UML tutorial] http://bdn.borland.com/article/0,1410,31863,00.html