Problem statement for the assignment/project

**Time Logging**

A software production company is given a task to develop a Time Logging tool for public sale to organizations in need of time control software for their employees. The company hopes that the TL tool will be in position to compete with the market leader – the tool called Time Logger from Responsive Software ([http://www.responsivesoftware.com/](http://www.responsivesoftware.com/)).

The scope of the TL project is delineated by the functionality offered by the Time Logger tool. The Responsive Software’s website ([http://www.responsivesoftware.com/](http://www.responsivesoftware.com/)) contains a detailed description of the Time Logger’s functions. Similar functions need to be present in the TL tool. The following points list the main functions.

The TL tool will allow employees to enter time records, i.e. time spent working on various projects and tasks and time without any work done (pauses, lunches, holidays, etc.). The time can be entered by directly (manually) recording the start and end times or by using the stopwatch facility. The stopwatch facility links to the computer clock and allows the employee to use start/stop command buttons to say when an activity started and ended.

The TL tool will allow identifying clients for whom the work is performed. Related functions are to bill the clients, produce invoices, and keep track of payments. Work expenses can be calculated using hourly rates and/or using fixed expense component. Of course, some activities recorded in the TL tool will not be billable to clients.
The TL tool will allow production of customized time reports with various reporting details suppressed or added, as required by the employee.

The TL tool will allow easy changes to time records already entered. It will also provide various sorting, searching and filtering capabilities.

Consider the function in which an employee uses the stopwatch facility of the Time Logger tool to create a new time record. This function is the responsibility of a use case subflow called “Create Time Record – Stopwatch Entry”. The GUI window supporting this subflow is shown below.

The Stopwatch Entry window is a modeless dialog box. This allows the user to access the time records in the primary window’s row browser, the menus and other features of Time Logger, while stopwatch is running. The display in the figure shows the stopwatch in the “Running” state after it has been started from the Stopwatch menu.

The window has buttons for starting/stopping the stopwatch. When stopwatch is running, the person can use the picklist fields and the Description field to fill in information about what he/she is doing. When the Stop button is pushed, the Time Logger adds a new time record in the row browser.

The Duration is calculated based on the contents of the Start, Now, and Pause Duration fields. The Now fields are not editable. The buttons Pause and End Pause control the pause duration.
The Reset button cancels the stopwatch without storing a time record in the database. The Hide button hides the stopwatch and makes it run in the background. The hidden stopwatch can be displayed again from the Stopwatch menu.

The iconic buttons plus, pencil and minus provide create, update, delete functions on corresponding picklists.

**Task 1** (? marks)

Using Rational Rose (or other UML modeling tool), develop a business use case model and a business class model for the TL software tool. Your use case model should be named Task1a and class model should be named Task1b.

Explain additional constraints in UML notes placed on the diagrams. Describe the meanings of the models, including any assumptions you have made, in text under the diagrams.

Note that in Rose (and most other UML modeling tools) you can create multiple class diagrams for the same model (project) but the diagrams are integrated and based on the same repository. This means that if you have a class, say Employee, then if you modify it in one diagram, the changes will automatically show up in other diagrams that contain the same class. Because in this assignment, you are asked to create three class diagrams (Tasks 1, 3, and 5) at different stages of development, one possible approach is that you name classes with numbered suffixes. For example, to have three versions of class Employee, name them Employee1 in Task1, Employee3 in Task 3, and Employee5 in Task 5.

Consider the following use case document for the use case “Create Time Record – Stopwatch Entry”. [Note to instructors: Creation of the use case document as below can be required to be done as Task 2. Accordingly, the next task is named Task 3]

**Use Case: Create Time Record – Stopwatch Entry**

**Brief Description**

This use case enables an employee to create a new time record using the stopwatch facility. In the stopwatch mode, the employee can track his/her time while working by starting the stopwatch (when the work starts) and stopping it (when the work is temporarily stopped or completed).

**Flow of Events**

**Basic Flow**

This use case starts when the user chooses to create a new time record by using Time Logger’s function to automatically time the employee’s activities while he/she is working. The use case shall start when the user selects a Stopwatch menu item from the primary window of Time Logger.

The system shall display the Create Time Record – Stopwatch Entry dialog box window as shown below. The display in the figure shows the stopwatch in the “Running” state after it has been started from the Stopwatch menu.
The Stopwatch Entry window shall be a modeless dialog box. This will allow the users to access the time records in the primary window’s row browser, the menus and other features of Time Logger, while stopwatch is running.

The window will have a button for starting the stopwatch and a button for stopping it. When the window is launched, the system shall enable the Start button and disable other action buttons. The system shall enable entry fields in the window.

The user can use the picklist fields and the Description field to fill in information about what he/she is doing. The user will push the Start button when he/she starts working on some task. When the Start button is pressed, the system shall activate the stopwatch recording of time.

At any time when the stopwatch window is available, the user can fill in the window’s fields, including time record Description. The user will push the Stop or Stop/Start button when the activity is completed.

When the Stop or Stop/Start button is pressed, the system shall add the time record to the primary Time Logger window. Ref. Alternative flow “Time overlap”. Ref. Postcondition “Stop button pressed”. Ref. Postcondition “Stop/Start button pressed”.

The Duration shall be calculated based on the contents of the Start, Now, and Pause Duration fields. Ref. Alternative flow “Duration invalid”.

The Now fields shall not be editable. The buttons Pause and End Pause shall control the pause durations (there can be many pauses within each timelog record).

The Reset button shall cancel the stopwatch without storing a time record in the database. The Hide button shall hide the stopwatch and make it run in the background. The system shall allow a hidden stopwatch to be displayed from the primary window’s Stopwatch menu.

The iconic buttons plus, pencil and minus shall provide create, update, delete functions on corresponding picklists (combo boxes).

Alternative Flows

Time overlap

The system shall not allow creating/updating a time record with date/time periods that overlap with any date/time period that already exists in the database for that employee.

Duration invalid

The system shall display an error message if the time Duration is invalid according to various business rules such as that the number of hours exceeds the maximum permitted for the employee.
Special Requirements

Legal constraints

The Time Logger software shall not impinge on any copyrights, trademarks and other ownership rights of Responsive Software – the vendor of the Time Logger.

PreConditions

The actor is authorized to access his/her time log.

PostConditions

Stop button pressed

After a time record is successfully created (saved to the database) as a result of pressing the Stop button, the control of the program shall go to the browser window, which highlights the row with that timelog record information.

Stop/Start New button pressed

After a time record is successfully created (saved to the database) as a result of pressing the Stop/Start New button, the control of the program shall stay in the dialog box, which resets all fields in anticipation that the next stopwatch timelog record will be started.

Extension Points

None

Task 3  (? marks)

Using Rational Rose (or other UML modeling tool), develop an entity class diagram (i.e. entity classes only) for TL corresponding to your use case document above. Your class diagram must be named Task3.

Show attributes in classes (specify both attributes names and Java types). Specify relationships between classes. Explain additional constraints in UML notes placed on the diagram. Describe the meaning of the model, including any assumptions you have made, in text under the diagram. Discuss alternative solutions, if applicable.

Task 4  (? marks)

Using Rational Rose (or other UML modeling tool), develop a sequence diagram and a corresponding collaboration diagram for the action of stopping the stopwatch, i.e. when the Stop button is pushed (clearly, this action must enter a new time record in the database).

The design must conform to the PCMEF architecture and the signatures of messages must be shown. The model must be explained in text. Discuss alternative solutions, and explain your choice of the solution.

Your sequence diagram must be named Task4a and collaboration diagram Task4b.

Task 5  (? marks)

Using Rational Rose (or other UML modeling tool), develop a class diagram corresponding to your sequence and collaboration diagrams from Task 4. Your class diagram must be named Task5.
Show attributes and operations in classes (specify attributes names and types, and complete operation signatures). Specify relationships between classes. Show visibility of operations and attributes. Explain additional constraints in UML notes placed on the diagram. Describe the meaning of the model, including any assumptions you have made, in text under the diagram.