SE 361: Object Oriented Design

Project Description

For this project you will consider the case study included in your textbook, which describes some automation needs for the "Just-the-Job" cleaning company. The manager of Just-the-Job has secured the funding for a new computer system to handle most of the paperwork involved in the company's daily routines, which are explained in Appendix B (pp. 321-323) of your textbook.

Functional Requirements

After an initial business meeting with the company's manager and you (the software developers) it has been decided that the system must ultimately provide the services described below. The services marked M are performed by the manager, whereas the receptionist normally performs those marked R. (Note that the manager may also perform receptionist functions if she chooses to do so).

1. **Manage customers (R & M)**
    a. Add a new customer (R)
    b. Delete an existing customer (M)
    c. Modify information stored about an existing customer (R)
    d. Display/print information about an existing customer based on his/her id or name (R)
    e. Display/print a list of all customers and their information (R)
2. **Manage employee (i.e. cleaner) (M & R)**
    a. Add a new employee—name, id, address, pay-rate, weekly schedule, etc. (R)
    b. Delete an existing employee (M)
    c. Display/print information about an existing employee (R)
    d. Display/print a list of all employees with their information (R)
    e. Modify information stored about an existing employee (R)
3. **Manage jobs/appointments (M & R)**
    a. Add/register a new job (R)
    b. Delete/cancel an existing job (M)
    c. Display/print the status of a specific job (e.g. job number; address of property, completion status, by-whom etc.) (R)
    d. Display/print a list of all jobs with their completion status (R)
    e. Display/print a list of all completed jobs (R)
    f. Display/print a list of all pending jobs (R)
    g. Modify information stored about an existing job (R)
4. **Manage customer invoices for <u>one-time</u> jobs (R & M)**
    a. Create a new invoice (R)
    b. Cancel an existing invoice (M)
    c. Modify an existing invoice (R)

d. Display/print an existing invoice (R)
e. Display/print a list of all invoices and their payment status (R)

5. **Manage customer invoices for <u>regular</u> jobs (R & M)**
    a. Create a new invoice (R)
    b. Cancel an existing invoice (M)
    c. Modify an existing invoice (R)
    d. Display/print an existing invoice (R)
    e. Display/print a list of all invoices for a given week with payment status (R)
6. **Manage customer payments (R & M)**
    a. Record a full payment (R)
    b. Cancel/credit a payment (M)
    c. Print receipt (R)
7. **Maintain manager's personal weekly schedule (R)**
    a. Add a new appointment (R)
    b. Cancel an existing appointment (R)
    c. Modify an existing appointment (R)
    d. Print weekly schedule (R)
8. **Save information (R & M)** Upon user request, save all data to disk at any time. (This is in addition to the automatic save to disk which occurs at shutdown)

Incremental Development

This project will be completed using three time-boxed iterations. A selected sub-set of the overall requirements will be designed, implemented and tested during each of the following iterations.

Iteration #1: (4 week time-box)

- Requirements to be completed: 1a,e 2a,d 3a,d
- Ability to save and re-load data between program-runs (automatic at program start-up/shutdown plus manual via "Save" menu option – Requirement 8.)

Iteration #2: (4 week time-box)

- Requirements to be completed: 1b-d, 2b-c, 3b-c & e-f, 4a-e

Iteration #3 (1 week time-box)

- You will design and implement some subset of the remaining requirements, and/or new or changed requirements, to be announced and specified later in the semester by your instructor.

Important Notes:

- For this project you are required to implement only a subset of the complete set of requirements described above. The key is, however, that your design should be such that implementing the full functionality would be straightforward.

- Obviously, the set of requirements for iteration 1 is smaller than that for iteration 2. The requirements for this iteration are the central requirements for the entire system. So, invest the time needed upfront in order to get the overall architecture and design right; this will pay off on subsequent iterations.
- Note that you will be given some additions/changes to the requirements later in the semester. (Changes like this are typical of real software development projects.) A well-designed (and agile) system will make incorporating these changes relatively easy.
- Remember that the goal of the project is to give you experience with developing software incrementally and iteratively and not so much in producing a robust industrial-strength software product.

PROJECT MILESTONES

Each iteration consists of a series of milestones. Each milestone is denoted by a two-part number e.g. milestone 1-2 is the second milestone for the first iteration. In each case, unless otherwise noted, the milestone pertains only to the requirements for that iteration, but some milestones include requirements that pertain to the whole system, not just the one iteration it is part of. (Please be careful to note which is which).

At the completion of each iteration you are required to upload all produced artifacts and present your work to the rest of the class by the due date specified below. The specific artifacts to be delivered for each milestone are described next.

Deliverables

0-0 (Upload the following deliverables by the due date posted on our course website)

- Upload on the course's web site a document with the following information:
  a. The name of your team and its members. You may work with a partner of your choice (or alone if you prefer; not recommended). Teams of three members may also be possible (if you cannot find a partner, email your instructor and he will find you one).
  b. Read carefully the above requirements 1-8. Then create and submit a list of any brief clarification questions you might have regarding these features. (Note: the goal here is to understand <u>what</u> is to be done; <u>not how</u> to do it).

1-1 (Upload the following deliverables by the due date posted on our course website)

- Use-case context diagram covering all requirements (not just iteration 1)—see figure 3.2 in your textbook.
- High-level descriptions for all use-cases in your use-case diagram (see figure 3.5 in your textbook).
- A project plan and time line for this iteration using a Gantt chart. You can use a project management tool such as MS PROJECT (links to some free tools are posted on our website)

1-2 (Upload the following deliverables by the due date posted on our course website)

- Expanded use case descriptions for only the requirements of this iteration 1 (see figure 3.6 in your textbook). Make sure to include both successful (typical event flow) and unsuccessful (alternate event flow) scenarios.
- A list of candidate classes derived from applying noun analysis of the problem description explained in Appendix B (pp. 321-323) of your textbook (see figure 5.3, page 123). Discuss which classes you decided to eliminate/reject as unsuitable and why.

1-3 (Upload the following deliverables by the due date posted on our course website)

- Detailed class diagram showing classes needed for iteration 1 including their attributes and relationships (see figure 5.15 in your textbook).
- CRC cards for the classes involved in the use cases done for this iteration (see figure 6.4 in your textbook).
- Sequence diagrams for the use cases you did for this iteration (see figure 6.13 in your textbook).
- Overall GUI design for covering all requirements. Include screen shots of all the forms, menus, dialog boxes, etc. to be implemented. Also, for each GUI form class you will create the TOE (Task-Object-Event) Tables as shown in class (see also example posted on our website).

1-4 (Upload the following deliverables by the due date posted on our course website)

- Unit tested code that implements the requirements of iteration 1 (information should be saved correctly on disk between runs)
- A Gantt chart showing your overall project plan, timeline and tasks for each team member (use a tool such as MS PROJECT)

- Team Review Meeting on the due date posted on our course website (instructor meets with each individual team for consultation)

<u>TEAM PRESENTATIONS FOR ITERATION #1</u> (due date posted on our course website)

2-1 (Upload the following deliverables by the due date posted on our course website)

- Extended use-case diagram covering all requirements (not just iteration 1) entailing any "include" or "extend" relations among use cases (see figure 3.9 in your textbook).
- Expanded use case descriptions for the requirements of this iteration 2 (see figure 3.6 in your textbook). Make sure to include both successful (typical event flow) and unsuccessful (alternate event flow) scenarios.
- Updated class diagram including classes needed for iteration 2 with their attributes and relationships (it is normal and expected to modify and fine-tune your class diagram from the previous iteration)
- An updated project-plan and time-line to include this iteration using a Gantt chart. You can use a project management tool such as MS PROJECT (links to some free tools will be posted on our website)

2-2 (Upload the following deliverables by the due date posted on our course website)

- CRC cards for the classes involved in the use cases done for this iteration 2.
- Sequence diagrams for the use cases and CRC cards you did for this iteration 2.

2-3 (Upload the following deliverables by the due date posted on our course website)

- Updated GUI design (e.g. new screen shots of all the forms, menus, dialog boxes, etc.) Updated TOE (Task-Object-Event) Tables as shown in class (see example posted on our website)
- Add tested code for this iteration to the code you wrote previously. (Of course, the code you wrote for iteration 1 should work correctly with this new code)

- Team Review Meeting on the due date posted on our course website (instructor meets with each individual team for consultation)

TEAM PRESENTATIONS FOR ITERATION #2 (Due date posted on our course website)

3-1 (Upload the following deliverables by the due date posted on our course website)

- Additional and/or modified diagrams needed to develop some new requirements (which will be announced later by your instructor).
- Updated project plan and time line to include this iteration using a Gantt chart.

3-2 (Upload the following deliverables by the due date posted on our course website)

- Additional tested code for this iteration.

- Team Review Meeting on the due date posted on our course website (instructor meets with each individual team for consultation)

TEAM PRESENTATIONS FOR ITERATION #3 (Due date posted on our course website)

Final Comprehensive Project Report (due date posted on our course website)

NOTE: only e-copies need to be submitted and uploaded

Logistics, Evaluation and Grading

- The above iterations are time-boxed with fixed due-dates. De-scoping of requirements is possible when well justified. However, all the stated requirements must be eventually completed and delivered by the end of the semester.
- You have considerable discretion in terms of the details of the user interface, database back-end, etc., as long as your project fulfills the requirements as stated.
- You will implement your system as a Windows application using VS.NET (with VB.NET or C#.NET).
- Also, all your documents should be produced using a word processor such as MS Word and diagrams should be produced using appropriate UML modeling or drawing tools such as MS VISIO or other similar tools (some free tools will be posted on our course's web site).
- At the end of each iteration you will give a presentation to the class describing briefly all the artifacts you have produced for every milestone.
- Your class peers along with the instructor, will evaluate you and give you constructive feedback and a score. This score is a measure of the completeness and quality of your work you are producing so far.
- Only your final presentation and comprehensive project report will be formally graded using a grading rubric that will be given to you later in the semester.
- Remember to maintain and upload on our course's website a complete electronic archive of all your work including documents, source code, drawings and other related artifacts.

- As part of your final deliverables, on the above due date, *each team member* must email the instructor with an assessment of how the team worked well (or not).