Homework #1 solutions

1.3 Should optimization be a focus of software engineering? Briefly explain, and justify your answer with a practical example.

Author Solution: The answer to this question may depend on the interpretation of "optimization." If it is about "optimization of software PQCT," then it is the focus of software engineering. If it is about performance optimization, then it should not be a focus, although SE also considers performance issues such as testing for performance. The database access example discussed in Section 1.5 is a practical example. Optimization could be a focus for a given project. For example, the construction of a compiler for multicore computers. In this case, it depends on whether the project is classified as a software engineering, or a computer science project. It might be an SE project. For example, it is constructed for a certain application. (See solution to Exercise 1.6 for more on optimization and SE.)

Professor comment: Basically I agree with this assessment, although it doesn't address one particular detail, namely that it's important to get the software to actually work first. In this sense, optimization may be a focus of SE and should be planned for as part of the project development activities, but should be relegated to a later iteration or release cycle. The rule of "brute force first, finesse afterward" applies in this case. It may also make sense to evaluate whether optimization is even needed for the project; if the application is small enough or simple enough, a brute force approach may be sufficient for operations, negating the need for a (possibly) expensive and time-consuming optimization effort which may add no business value. Additionally, some optimizations may be based on specifics or idiosyncrasies of the programming language or environment, which can cause maintenance headaches in the later life of the application, or may even prohibit some desired updates which the language/environment won't support.

1.4 Identify three computer science courses of your choice. Show the usefulness of these courses in the software life-cycle activities.

Author Solution: An Algorithms and Data Structures course is useful in the implementation phase for the design and implementation of algorithms and data structures to implement business processes. In particular, the course lets the student know the available algorithms and related data structures. The computational complexity lets the student select appropriate algorithms and data structures according to the nature of the computation.

A Database Systems course is useful in the analysis, design, and implementation phases. In the analysis phase, it helps the student understand database related requirements such as the need to support multiple databases for some applications. In the design phase, the course enables the student to design the database to fulfill the requirements and constraints. In the implementation phase, the course provides the student abilities to store and retrieve information with a database.

A Discrete Mathematical Structures course is useful in many phases of the life cycle. In particular, mathematical logic lets the student design and implement logically consistent and logically complete algorithms, and check for such properties during design review and code review. Graph theory helps the student understand design diagrams such as UML diagrams because UML diagrams are directed graphs. Thus, concepts and algorithms of graph theory can be applied. Examples include fan-in and fan-out of a class, transitive closure for computing the change impact of a class, traversal algorithms for calculating reachability in a state diagram.

Courses on programming languages are useful for implementation, testing, and SQA activities. Network courses are helpful in SE project that must communicate with a remote computer, such as accessing a remote database. Artificial intelligence courses are useful for SE projects that involve heuristic, and/or learning algorithms.

Professor comment: I have nothing to add to this solution.

1.6 What are the differences and relationships between OO software engineering and conventional software engineering? Discuss whether OO software engineering will replace conventional software engineering.

Author Solution: The main difference between conventional software engineering and OO software engineering is paradigm shift – that is, how they view the world and systems. Because of this, they differ in the basic concepts, basic building blocks, and starting point for the conceptualization, design and implementation of software systems. These in turn affect SQA, project management (for example, effort estimation and planning), and SCM. Will OO replace the conventional paradigm? The answer should be no. (footnote: A student's solution may indicate "yes," and provide convincing arguments. Such a solution should also be considered a good solution.) The reasons are:

1. There are numerous systems that were developed using one or more of the conventional paradigms. It is very costly and risky to replace these systems. Therefore, the other paradigms will continue to exist because bug fixing, performance improvements, and functional enhancements to these systems are required.
2. There are hundreds of thousands of organizations and millions of software developers using only the conventional paradigms. It is practically impossible and unjustifiable to require them to convert to the OO paradigm.
3. A conventional paradigm may be more suitable for some projects. For example, scientific computing typically involves series of transformations of input into output. Therefore, the function-oriented paradigm is more suitable for such applications. Moreover, scientific computing emphasizes computing speed, the ability to solve complex computation problems, and the accuracy of the result. OO programming languages may not satisfy such requirements. These and the facts that scientific computing is there to stay and expand into computational sciences imply that the function-oriented paradigm will continue to exist.

In addition to the above, one should know that different parts of a system may be developed using different paradigms. For example, a subsystem that performs scientific computing may be developed using the function-oriented paradigm. A database subsystem may be developed using the data-oriented paradigm. In practice, there are systems that are modeled and designed using the OO paradigm but implemented in a non-OO language. Similarly, there are projects that are modeled and designed using a conventional paradigm but implemented in SmallTalk or C++.

Professor comment: I have nothing to add to this solution.

2.1 What are the similarities and differences between the conventional waterfall model and the Unified Process model? Identify and explain three advantages and three disadvantages of each of these two models.

Author Solution: The waterfall model and the Unified Process (UP) model are similar in the sense that they are process models, they define phases, the activities and products of each of the phases. The waterfall process is a sequential process although backtracking is possible. The UP, on the other hand, is an iterative, incremental process.

Waterfall process advantages are: (1) it facilitates project management, scheduling and status tracking, (2) it can be used for function-oriented team organzation, and (3) it is more appropriate for some types of software project. Its disadvantages are: (1) it is difficult to respond to requirements change, (2) the long development duration is unacceptable, and (3) users cannot experiment with the system until late in the development life cycle.

UP advantages are: (1) its iterative process can better accommodate requirements change because changes can be made to remaining iterations, (2) it is use-case driven, allowing the development team to focus on customer value – that is, development and deployment of high-priority use cases as early as possible, (3) it is incremental, this reduces the risk of requirements misconception. It disadvantages are: (1) an iterative process is more difficult to manage and schedule, (2) the early versions of the UP emphasize too much on documentation and much of it is not used, (3) the UP is a process, not a methodology, therefore, it is useful only for experienced software developers.

Professor comment: I have nothing to add to this solution.

3.1 Provide a brief description of the functions of a vending machine. Identify and formulate all functional and performance requirements.

Author Solution: A vending machine (VM) allows an operator to program the bill acceptor, the coin acceptor, and vending prices. A customer can vend items with the VM, receive the vend item and the change. The VM shall keep track of the all-time total amounts and since-reset total amounts of bills and coins received, respectively. An operator can reset only the since-reset total amount. The operator can also view the all-time total amount and the since-reset total amount.

The functional requirements of the VM include the following:

Software Functional Requirements

R1. The VM shall allow an operator to perform the following VM operator functions:

- R1.1 An operator shall be able to program the bill acceptor for bills accepted ($1, $5, $10, $20).
- R1.2 An operator shall be able to program the coin acceptor for coins accepted (nickel, dime, quarter, and dollar coin).
- R1.3 An operator shall be able to view the all-time total amount of bills and coins received.
- R1.4 An operator shall be able to view the since-reset total amount of bills and coins received.
- R1.5 An operator shall be able to reset either or both of the since-reset total amount of bills and since-reset total amount of coins received, respectively.
- R1.6 An operator shall not be allowed to reset the all-time total amount of bills and coins received.
- R1.7 An operator shall be able to program the vending prices for the vending items using the alphanumeric keypad mounted behind the front panel.

R2. The VM shall allow a customer to vend items.

- R2.1 A customer shall be able to vend items with bills.
- R2.2 A customer shall be able to vend items with coins.
- R2.3 A customer shall be able to vend items with a combination of bills and coins.
- R2.4 The VM shall display the total amount of bills and coins inserted by the customer for the current transaction.
- R2.5 A customer shall be able to press a letter key and a digit key on the keypad mounted on the front panel to vend an item when the total amount of bills and coins inserted is sufficient.
- R2.6 The VM shall dispense the change and the item selected by the customer if the inserted amount is sufficient and the selected item is available, else the VM shall display an error message.
- R2.7 A customer shall be able to press the return key to cancel an outstanding transaction. The VM shall return the funds inserted by the customer.
- R2.8 ...

Hardware Functional Requirements

R3. The VM shall include all hardware components to support the VM functions:

- R3.1 The VM shall include a programmable bill accepter, and a programmable coin acceptor.

R3.2 The VM shall include an operator keypad for the operator functions. The keypad shall be mounted on the back of the front panel and accessible only when the front panel is unlocked.

R3.3 The VM shall include a customer vending keypad, mounted on the front of the front panel.

R4 ...

Nonfunctional Requirements

Nonfunctional requirements for the VM include performance, response time, user interface (UI), and other nonfunctional requirements. For example, the VM shall display the total inserted amount, and dispense items within X and Y seconds, respectively.

Professor comment: Several of these requirements need some work. For example, R1.7 needs a "should" or "will" to explain more about what is meant by "behind the front panel." R2.5 needs to be broken down to add requirements for verifying the amount; different items will cost different amounts, so the amount of money inserted needs to be verified as sufficient after the customer selects an item. R2.6 should become two requirements. R2.7 should become a requirement and a "will" statement. R3.1 should become two requirements. R3.2 second sentence should become two "will" statements.

4.1 Produce a software requirements specification (SRS) for a library information system that is similar to the system in use in your school. At the minimum, the system should provide functions to allow the patron to search, check out, and return documents, respectively.

Author Solution: This exercise is quite common. There are quite a few solutions published on the web. Moreover, the library information systems are different for different universities. Therefore, a solution is not provided.

Professor comment: What a cop-out…

The SRS should follow the samples that are provided on the class web site. These samples are modeled from industry standard documents, which in turn follow the DoD standard which nearly everyone uses; the sections of the document are well-defined in those exemplars.

Requirements in the system should include (but are not limited to) the following items:

- add a resource (book, video, etc.) to the collection
- remove a resource (book, video, etc.) from the collection
- check out a resource (book, video, etc.)
- check in a resource (book, video, etc.)
- locate a resource (book, video, etc.)
- add a library user
- remove a library user
- suspend a library user
- reinstate a library user
- add a librarian

- remove a librarian
- suspend a librarian
- reinstate a librarian
- renew a resource (book, video, etc.) for a specified time
- calculate late fee for a resource (book, video, etc.)
- levy a calculated late fee on a library user
- notify a library user of any overdue resource (book, video, etc.)

5.2 Produce a one-page description of the business operation for a hotel reservation system. State practical and reasonable assumptions.

Author Solution: The description of a hotel reservation system is the following:

"The hotel reservation system (HRS) is intended to be an online reservation system. Authorized hotel employees shall be able to enter and update hotel and room information online, including special offers. Potential guests shall be able to search rooms using a number of search criteria. The HRS shall respond with a list of summary descriptions of the rooms that satisfy the search criteria. The guest shall be able to view the detailed description and photographs of any of the rooms. The guest shall be able to select several rooms to view a comparison of their features.

The description of the hotel includes the address, phone number, fax number, email address, facilities such as cafeteria, bar, conference rooms, swimming pool, recreational facilities, directions to the hotel and a map, among others. The description of a room shall include the room number, room size, bath room (shower or bath tub), television, Internet connection, number of beds and bed size, and availability, among others. The charges of a room may vary according to the travel season and may be adjusted manually as well as automatically (e.g., automatically increase by 5% each year, or according to changes of price index). A guest shall be able to reserve rooms with reservations. A reservation may reserve one or more rooms for a period specified by the guest, provided that the room is not reserved during that period of time. A confirmation message shall be sent to the guest's email account. The rooms are reserved only to 6:00 pm for the guest on the first day of the reservation unless the reservation is guaranteed by a nonrefundable first night charge to a credit card provided by the guest. That is, the amount of one night charge for each room of the reservation is charged to the credit card after 6:00 pm of the first day of the reservation. When a reservation is made, the room becomes unavailable for the period of time specified by the reservation. A guest is allowed to update or delete a reservation, provided that the reservation is still outstanding. Other services may be added to this project. For example, confirmation and notification services can be selected by the customer, who may specify the means (such as email, text message, or phone call) to deliver the services."

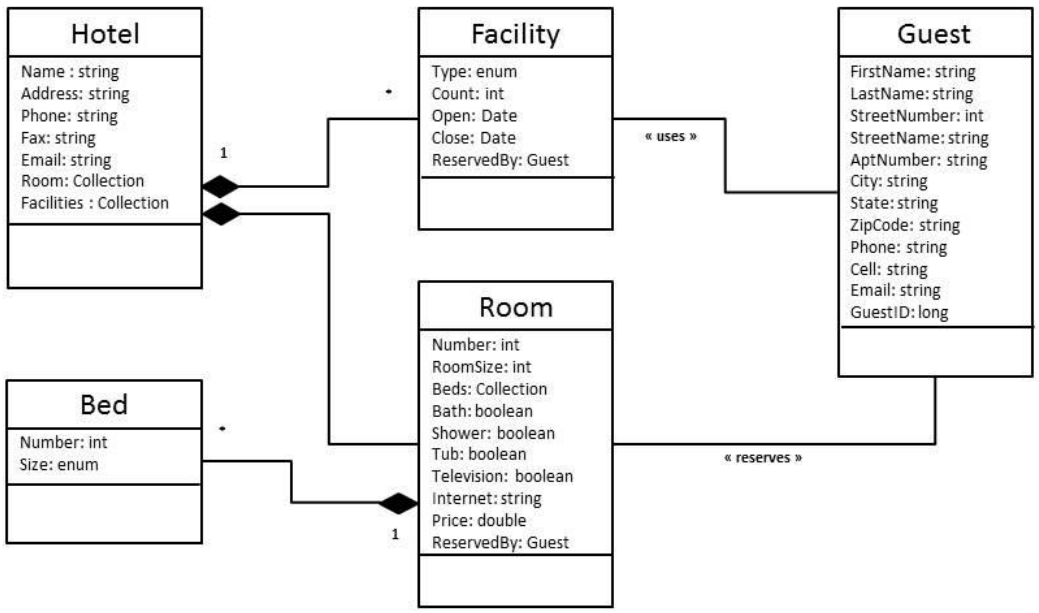
Professor comment: I have nothing to add to this solution.

5.5 Draw a UML class diagram as a domain model based on the classification results produced in exercise 5.4.

Author Solution: The conversion of the classification result to a UML class diagram is straightforward and is omitted here.

Professor comment: another cop-out…

Here is my take on a class diagram, based on the written description from the answer to exercise 5.2 and the brainstorming of classifications from exercise 5.4:



The aggregations show the relationships between the hotel and the rooms and facilities that it has. A case could be made that a "facility" is a subtype of a "room", but I elected to keep them as separate classes. Either way, the rooms and facilities are part of the hotel, and the diamonds are black because the rooms and hotel don't exist without each other. The same logic is applied to the room and bed classes. The Guest class "uses" the facilities and "reserves" the room, as indicated by the stereotypes.