CS-540/485 FALL 2015

MID-TERM EXAM: 65 Points

DUE BY 4:00 P.M. 10/24/2015

******** SUBMIT **PDF** VIA BLACKBOARD *********

NAME: **TAMER AVCI**

STUDENT ID: **2105024**

DIRECTIONS: Each question is worth 10 points total. These are short answer or short essay questions. Your answers should be brief but complete. The exam covers chapters of the textbook that were subject of lectures, plus lecture materials and readings. The slides are available on the course website for review of the textbook materials.

TIME: The completed exam must be submitted via Blackboard by the due date and time above. Please measure the time it takes to complete this exam if it was completed in one sitting. Late submissions will be accepted with a 5-point penalty 24 hours after the deadline. After that, no further submissions will be accepted.

SAVE THIS DOCUMENT WITH YOUR ANSWERS AS A PDF. Submit the PDF to Blackboard at the Midterm submission link/folder.

1. Agile methodology (10pts total)

(A) During the last 10 years, Agile Methodology has become a very popular software process model. In your opinion, what are the top 5 features of Agile that are different from the traditional software engineering approach and make it so popular? Explain your reasons for selecting them. List 3 weaknesses of Agile you think makes it less desirable? You can look at the 2014 Agile Survey by Version One on your class documents on Blackboard, under otherReadings, file AgileStats.pdf. (5pts)

i. **Ability to manage changing priorities:** After initial planning you can always change the priorities as you get feedback from the client. In fact, it is expected from the client to make and offer changes to the product development.

ii. **Ability to keep the program up to date:** Because it is easier to add features to the software at all times, latest developments in the industry will always be implemented, thus project time and deadlines will not hinder product quality.

iii. **Continuous testing:** Because testing is being done throughout the process, bugs are caught and solved in the cycle immediately. Issues will not altogether rise at the very end.

iv. **Project visibility:** With Agile development, product could be launched and run at any point in the cycle so that clients can see the progress, adjust their expectations and always get what they need at the actual launch date.

v. **Collaborative environment:** Agile development is especially teamwork oriented environments. Each developer and designer work on different modules and then they work together to integrate all of them into a cohesive software. This is beneficial because it avoids incompatibility of different modules due to collaboration from early on.

Weaknesses---

i. Although Agile process is highly flexible, it lacks the structure that the waterfall process has. Without a definitive and concrete plan from the beginning, everything remains a bit vague.

ii. Because Agile requires collaborative environment and testing all modules throughout the development, it is for sure more time consuming than other models. This also means that everyone needs to be committed and motivated for the development. If a project team loses someone in the middle of the project it could lead to a catastrophe as it is very person-based.

iii. Agile projects are hard to predict especially when it comes to timelines and budgets. Because feedback is endless and provided at all times, deadlines may not be met and budgets may be exceeded.

(B) Explain why you think Agile is (or is not) an appropriate process model for your class project. Construct a convincing argument that includes how stand-up meetings are or are not useful, and what other features may be needed. (5pts)

There are fundamentally two reasons why Agile is more suitable and appropriate to the projects we have for this particular class. Firstly, the projects heavily need client input because the goals are not well defined and the outputs need more than just improvements. Often times, new suggestions are introduced and there are far better ways to address certain aspects of the already existing products. Therefore, continuous interaction is a must between the client and student groups.

Secondly, because we need to develop intermediate results, waterfall method would prevent us from having products and testing them at any point during the semester. This is needed because if the product developed within waterfall method did not quite turn out to be the product that the client wished to have, there is no more time to change that and improve it due to time constraints in a single semester.

Having stand-up meetings or scrums significantly contributes to these two reasons. They provide touch-base with the team, updating each team member how the team will move forward and how the priorities should be changed on a regular basis. This proves to be useful as it allows us to see the process clearly and be more proactive in the development.

Another feature that might be added to the development could be more strict deadlines and perhaps enforcing certain days solely dedicated to implementation and putting all the ideas into real code. Doing them over time works too, however it is not completely time efficient.

2. Version control (10 pts total)

(A) Describe 4 key features of version control systems and explain why they are indispensable in today's software development. (5 pts)

a) Being able to fix your mistakes and revert to previous versions of code.

b) Being able to see the changes between two or more versions of your code.

c) Being able to easily work with multiple people on your code.

d) Being able to see how much work is being done, where and by whom.

All these features provide a crucial framework, where it is incredibly easy to keep track of every single change on the software, to safely experiment a new feature without interfering with working master code and to easily collaborate with multiple people knowing every single person's contribution to a particular software. Today's software development is never individualistic and always changing and the only way to keep up with the latest developments and to do it securely is through version control.

(B) Give all the necessary commands in the order you need to run them to update a file "myFileVersion" from your own project repo on Github. Assume your teammates have already changed this file in the repo several times in the past. Give all the commands for each of the following cases:

1) nobody has checked the file out, after its last update (2pts)

```
Git pull
//update your file
Git add myFileVersion
Git status (to check if it's on local repo – not required)
Git commit –m "I changed xx"
Git push
```

2) your colleague Joe is working on it but didn't update his version yet (3pts)

```
Git pull
Git checkout –b newbranch
//update your file
Git add myFileVersion
Git status
Git commit –m "I changed xx while Joe was working on his code"
Git push
Git merge master
```

3. Knowledge questions (10 pts total)

(A) What is refactoring and when is it needed? (2 pts)

Refactoring is a technique for improving the design of an existing working code by applying behavior-preserving transformations. The cumulative effect of each change is significant and leads to clean and extendable code.

Just because there exists a working code does not mean it is perfect. Often times, the code written is far from perfect. The software is usually not easily maintainable or extendable despite the fact that it does the job. Given that code is read more than it is written there is a need to polish the code, perfect the design without changing its functionality so it is easier for incoming people to read and understand your code and easily make changes as domains transform and new technologies are introduced.

(B) What is a User Story and what are its weaknesses? (2 pts)

A user story is a tool in agile software development. It is a description of software feature from an end-user point of view. It describes the type of user, what they want, why they want it and how they like to use that feature. A user story helps to create a simplified requirement. However, one weakness is that user stories feel informal, and put lot of emphasis on human-to-human interaction to get the details. By doing that teams may lose some technicality or overlook certain technical aspects. Complicated features may not be well described by user stories.

(C) Describe 4 key phases of a software process model.(2 pts)

Waterfall model:

This model has 5 phases. 4 of them are: Communication, planning, modeling and construction.

In the communication phase software requirements are to be gathered by laying out functional and non-functional features. This phase usually includes a set of use cases that describe the interactions between the software and the end users. This establishes the basis for an agreement on what the software is and is not expected to do.

Planning follows communication in that each set of requirement is estimated and scheduled. They should be documented, actionable, measureable, testable and traceable related to identified needs for the design of the software. Planning guides the team throughout the process.

Next comes the modeling phase. This phase constructs the architecture and intends to conceptualize and frame complex systems. Design phase usually creates a solution to a problem in hand with available resources. It focuses on the capabilities and there can be multiple designs for the same problem. One of the most important things to understand is that design is not coding. Detailed procedural processes can only be designed with a certain level of abstraction which would be higher than the source code.

Design is followed by coding. This is where the software actually gets developed and integrated. Implementation follows the design and requirements. Source code is created in this phase and tested after to systematically trace and fix the bugs.

(D) Describe a "scrum" (stand-up meeting) and explain its purpose. (2 pts)

Scrum is a standup meeting used in agile software development that brings the team together to discuss the progress, issues faced and what is going to be handled next. The scrum meetings are not used as a problem-solving or issue resolution meetings. Problems are solved offline and usually dealt with the relevant group immediately after the scrum. During the scrum meeting each team member answers the following:

What did you do?

What will you do next?

Are there any impediments in your way?

After having an excellent understanding of what work has been done and what work remains, team members make commitments to each other on what they are going to complete until next scrum meeting. Therefore, scrum ensures continuous progress on the project and establishes common platform to see the goals of each member clearly and precisely.

(D) What is aspect-oriented development? List four cross-cutting aspects in your project (2 pts)

Aspect-oriented programming focuses on certain functionality or aspects than can be spanned across the entire code. By using AOP you can modify your aspects to meet new requirements, allowing an application to adopt to new characteristics as it develops. You can furthermore abstract the cross-cutting aspects to achieve a dynamic application by defining specific places in your code where the cross-cutting code should be applied. This introduces new properties into objects without the objects having that knowledge before. In our project we have the following four cross-cutting aspects:

1- Drag and drop behavior on all grid overviews (design pattern)
2- Authorization of the user
3- Extracting data into csv
4- Placeholder behavior as a sidebar

4. Design and requirements modeling (10 pts total)

(A) What is the main purpose of object-oriented (OO) design and programming and why did it become so popular in the software world? (5 pts)

To understand the object-oriented programming one needs to look at an object first. Objects are discrete bundles encapsulating functions and procedures that are all related to a particular real-world concept such as a bank account or a car, or even a car engine. Other pieces of the code can access the object only by calling its functions. Object-oriented approach may therefore be called an abstraction paradigm whose purpose is to organize a program's structure so that one can build programs using these abstract models called "objects" that bring their own data and behavior into one unit. This allows one to work with code that is a little more connected to the actual problem space and makes the programs more readable and manageable through individual chunks of models or objects. This was the very reason that OO became so popular in the software world. Before OO, programs were just one huge sequential flow and they were not so much readable and if there was some big problem to solve then programmers needed to code in a hectic way as they could not form any groups.

(B) How do the requirements modeling steps described in your textbook map onto the work your team has done for your project? Do you think this was efficient? Propose an alternative requirements modeling approach that could have worked for your project (5 pts)

Requirements described by the textbook translated into our project in the form of user stories. This proved to be efficient to a certain extent. It was quite easy to go from a simplified user story to the implementation idea in a short period of time, however certain difficulties also came up due to lack of extensive and detailed descriptions e.g. what to do if a certain class period was blocked or overlapping.

Because of these edge cases, I believe use cases requirements model could have worked for our project too. An example use case would include name of the case, a short description, preconditions, postconditions, a basic course of action and then alternate courses. This approach sums up everything that might occur from one case in a very nice way. In the preconditions section we have our assumptions and basic pre-settings that must be given in order to begin the case. Postconditions describe the state we will be in if case is completed. After a user-story alike basic course of action we have got alternate course of actions in which the navigation in the software is explained if the user does not follow the conventional way. Because our project also has a good amount of nested scenarios due to the very nature of class scheduling, this requirements model would allow us to see the scenarios in one snapshot.

5. Read the following short article by Joel Spolsky, and then discussing it. (10pts)

“Duct Tape Programmer” by Joel Spolsky praises pragmatism in the software development world that is embodied in a “duct tape” programmer. Spolsky emphasizes how getting something out there is the most important thing and all else could be dropped in favor of delivery of the software.
It further illustrates the idea that pretty code, advanced frameworks, patterns or tools can never be an excuse for not delivering your product.

I understand Spolsky’s stand as far as delivery’s importance goes because no matter how much you accomplish during the construction of the software or how advanced you make your software to be, it will be of no value if you cannot deliver or finish the product because that is, after all, what the client needs and none of the clients will be happy unless you ship your software.

That being said, there are a couple points where I disagree with Spolsky. “Duct tape programmers” or “duct tape programming” may work in several instances but because it is not put into rigid testing, it might fail in several other cases that are not as common on a daily basis. This could be a working trade-off in certain domains, but if we think about the air traffic control systems for example, those systems are the ones that cannot afford to fail. Thus, putting up a duct-taped program up and running in this kind of domain may lead to huge problems in case they crash.

Furthermore, agile development does not necessarily put quality in front of delivery. The process constantly tests the modules, so modules are always ready to launch. They are also backed up with the advanced technology as agile is iterative. And if deadlines approach quickly, agile development teams can adjust themselves well in advance to make sure they prioritize delivery without sacrificing quality.

6. Interface design (15 pts)

(A) How did you perform the 'interface analysis' of your own project? To answer, you need to address each one of these three 'understanding' steps: user, task, concept. For each of these steps provide 3 questions (your choices) and their answers that you consider as best to help for designing the interface of your project. For each, briefly state how the answer reflects in the project interface (e.g., a flower shop (as user) would like to see flowers on their interface, along with pictures of their flower arrangements and their prices (task: sell flowers)). (9pts)

For the interface design we asked the following questions:

(User) Who will be using the class scheduler- students or faculty or both?

- Only faculty and MATHCS department staff will be able to access and use it. The software's sole purpose is to help the department schedule the math/cs classes for a given semester. Main user will be Erin Nagle with faculty overseeing the status and changes.
- Our software takes into account that we have one main user to work with the scheduler so we tailor to interface for her needs as much as possible. This includes for instance a placeholder area for her to drag&drop classes easily.

(User) How often will the software be used?

- The software will be used prior to each semester when it is time to set up a time schedule for every class offered by MATH/CS department.
- This reflected in our project in that the scheduler app could only be used to accommodate class events and no any other events.

(User) What are the consequences if a user makes a mistake using the system?

- The application should permit overlaps, no consequences.
- The system would allow for mistakes in that the user can put classes in the same time slot. We designed the interface that it handles exceptions such as overlaps.

(Task) How can a user perform adding a class?

- A user can add classes by clicking on add class button and specifying the class details.
- Basically, the interface allows the user to add classes without leaving the home page. We designed our interface so that everything is happening in place

(Task) How can a user get an overview? What tasks will be performed meanwhile?

- The user can switch between three main interfaces. One that displays MWF classes, one for TuTh and finally one for the entire week.
- This question helped us design the interface in such a way that it gives the user full control as to what he/she wants to see and does not want to see e.g. getting rid of certain days to allow user

focus on a smaller time frame, and vice versa to focus on the big picture.

(Task) What is the sequence of work tasks—the workflow?

- This question enables us to organize the workflow so that it is very easy to keep track of and prevents the user from getting lost. The workflow in our project is as follows:
  User opens up the scheduler, (1) inspects the grip. It has two choices at this point:

a) Add or remove class
b) Change a certain class
   Go back to (1)
   For a) a new class will be added/removed on the grid. The user will then drag and drop it to a specific location.
   For b) the user will drag&drop a certain class that he/she wishes to change.

(Concept/Content) Will mechanisms be available for moving directly to summary information for large collections of data?

- This question completely overlapped with what the client requested us to do. They had a feature in the interface to summarize the calendar information into a csv. File which was integral to our interface analysis. We will keep the same feature.

(Concept/Content)How will color to be used to enhance understanding?

- Color codes enhance the user experience in our project in that different colors will represent different classes. All blocks indicating the same class will be in the same color.

(Concept/Content) Can a user alter the grid's structure?

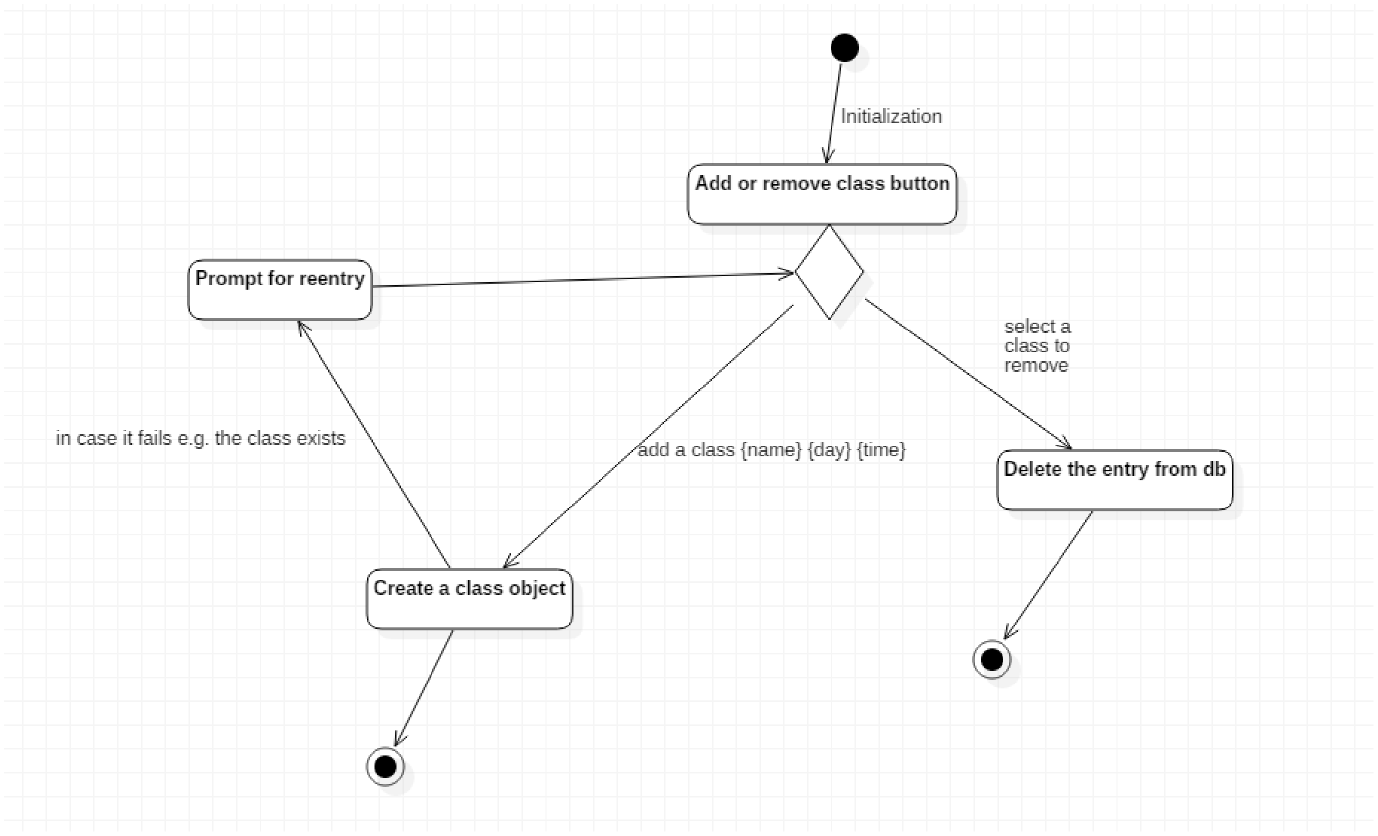
- No. Because the grid has dimensions that are already set before, all the blocks in the grid would have to be changed as the user changes the grid. This would complicate the interface so much that the user would have to carry a significant amount of workload that is adjusting blocks and losing autonomy.

(B) Pick an interface design pattern that you are using in your project and explain why do you think it is a good choice (or not) for the project. Provide its reference from the patterns library. (3 pts)

One design pattern that we are using in our project is the drag & drop interface on a calendar grid. I believe it was a good choice because one of the major problems that the client facing was the need to re-arrange the layout of modules on a web page directly with the mouse. Drag and drop allows the user to take a certain class from its set location and move it to wherever she wants without the hassle of actually going somewhere else to manually edit the time & location. This also served our main interface design goal that the user should not leave the page that she is working on to change something on that very page.
Reference: Yahoo design patterns > rich interaction > drag and drop > modules

Continued on the next page..

(C) Provide either a flow of activities diagram or a flow of interaction diagram of your project. You can chose to show the diagram corresponding to a specific scenario. You should use UML to build the diagram. (3 pts)



Enjoy!