CS 5744: Software Design and Quality

FINAL EXAMINATION

Due: **11:59 PM, Monday 12/12/2022**, on Canvas

100 points

Name: _Gasser Ahmed_____

INSTRUCTIONS:

This examination is a "take home," open book, open notes test. Please write up your answers in this file *in the space provided*, generate a PDF version named "final-*yourPID*.pdf", and upload it to the corresponding Canvas assignment by the deadline shown above. Note the page limits for answers that appear in each question. Please **do not change** the font, type size, or margin settings in the file.

Because of the open-book nature of this examination, you may use any information sources at your disposal (the course text, other textbooks, the web, journal articles, etc.). However, you **may not ask any other person for assistance** in answering any examination questions (whether or not they are involved in the class). Any quotations or significant ideas in your answers that are taken from other sources **must be cited**. A section for references has been added at the end of the examination for this purpose, although you should feel free to leave that section blank if all material is entirely your own. There is also no need to specifically cite references mentioned in the questions themselves.

Virginia Tech Graduate Honor Code Pledge

I pledge that this examination has been completed in compliance with the Graduate Honor Code and that I have neither given nor received any unauthorized aid on this examination.

Signature: _Gasser Ahmed_____ (sign or type your name to complete the honor pledge)

1. Defining Software Quality

1. [30 pts., 1 page limit]
   Several definitions of software "quality" have been discussed in class. Answer the following questions about software quality:

   a) Why is there no "universal," generally agreed-on, list of quality attributes?

      I believe there is no universal list of quality attributes because of how each author or architect has their own point of view of what quality means and what the product should provide to its users. Some believe that a quality product should only care about its substantial benefit to its end users while sacrificing reliability or performance while others believe that performance and the user's benefits should go together side by side. In addition, I believe the purpose or scope of a project plays an important part in defining that list of attributes. For example, a list of attributes will not be the same for a public customers' app like Doordash (where customer satisfaction is the main priority) as a company's internal app like microservices apps (where performance is the main priority).

   b) Why is there no single, generally accepted measure of software quality?

      I believe there is no single accepted measure of software quality because they are usually based on assumptions which are not standardized and may depend upon tools available and working environment which might be different from one project/organization to another. In addition, most of the software quality measures rely on estimates of certain variables which are often not known precisely. Lastly, projects/organizations usually have different ideas, service, and goals that would lead to a difference in the software quality measures used for each project. For example, measuring reliability or the mean downtime of a simple app like personal notes is redundant work and useless because there is nothing bad could happen if that app was down, while measuring that same attribute for an air traffic system is mandatory because people's lives are dependent on such system's reliability that makes its downtime unacceptable.

   c) Why are "indicators" used instead of measuring quality attributes directly?

      I believe "indicators" are used instead of measuring quality attributes directly because quality is a concept that varies from one person to another which makes it too vague to measure directly. In addition, quality is intangible and multi-focus, so we use indicators to give us a more meaningful measurement instead of measuring it directly by unpacking its attributes and making each one of them observable. Furthermore, indicators come after a thorough analysis of both the product/process (like programs and documentation) and the attribute itself (like maintainability) so its relationship to such attribute is undeniable and based on true evidence which makes it more accurate. Lastly, indicators can be based on multiple factors to provide a more effective and reliable indicator which accordingly allows the measurement of attributes that usually can't be measured. For example, if we want to measure the testability attribute of a product, indicators allow us to measure that attribute by basing our indicator on both the number of test cases and the length of time that each test case takes to complete.

2. Software Architecture

2. The papers we have read describe software architecture and a number of architectural styles. Using "software architecture" as an explicit part of a system development effort and choosing a specific architectural style in guiding your system design choices provides both advantages and disadvantages.

   a) [10 pts., 1 page limit]
   Identify the **most significant advantages** conveyed by choosing one well-understood architectural style for your software design. Justify your answer.

   I believe choosing one well-understood architectural style for your software design enhances understandability and readability of the software in the sense that the architecture itself is a standard for writing code and is a requirement for developers on how to write and where, so the software code becomes of the same type, which simplifies further product support. As a result, when a new developer joins the development team or takes on an existing project, they will be able to navigate the project structure and start developing much more quickly and with more confidence that new features can be added without any issues.

   In addition, it also enhances the software maintainability because when new features need to be added, it is very clear which part of the architecture needs to be modified and developers can work together on different layers of the architecture at the same time. For example, if we have an online shopping website where it uses the client-server architecture and we need to add a product review feature, some developers can work on the client-side of it developing and styling the review rating elements while other developers can work on the server-side developing the database connection between the review database tables and the backend code, which leads to a faster and more effective development process.

   Moreover, with a well-understood architecture in place, it is easier to divide the software system into smaller and more manageable modules that helps in introducing microservices architecture when scaling up operations which also improves the reusability of the system.

   Furthermore, if the chosen architecture is well understood and detailed during the development phase, it will significantly reduce the chances of logical errors in the software code which accordingly reduces the possibility that the code does not work as expected considering that requirements and functionality of the project have not changed in the development process.

   Lastly, having a well-understood architectural style for your software design helps in preventing schedule and budget outruns by discussing and keeping high-impact factors into consideration that can affect the development, deployment, and maintenance cycle. Additionally, choosing an architecture requires a thorough analysis of the software behavior before it is actually built which allows the development team to provide stakeholders with the expected outcome of their requirements resulting in mitigating risks and saving costs that could go in building or changing parts of the solution.

   That is, a well-understood architecture helps in maintaining the quality attributes of the software throughout its lifetime and makes it profitable in the long term because of how it becomes easier to scale and evolve which leads to saving developers' time while also serving the customers' changing requirements.

2. Software Architecture

[10 pts., 1 page limit]

At the same time, Shaw and Garlan acknowledge that most large systems do not cleanly follow a single architectural style, and instead use different styles in places or use hybrid styles. Explain why complex applications seem to defy the use of just one architectural style. Justify your answer, and also reconcile it against the advantages provided by using a single style that you described in part (a).

I believe complex applications usually use more than one architectural style because of the emergent behaviors of those applications that occur as a result of the interactions, dependencies, and relationships that form when that system's individual components are placed together, which requires multiple architectural styles to merge together to cope with those behaviors emergence and take the best out of each architecture that matches with the characteristics of that system's components to get the maximum benefits of each architecture and limit the effects of each system architecture's risks.

For example, if we have a complex system like an online file approval system like DocuSign, where the system relies heavily on event-based scenarios (like if a document is completed, send an email to the approver), in my opinion, the most suitable architecture would be using the following three architectures: Event-Driven, Shared Memory, and Pipe-and-Filter architectures, where Event-Driven enables asynchronous events to trigger and communicate between decoupled services, Shared Memory enables multiple users to exchange information by writing to and reading from pools of shared memory, and Pipe-and-Filter enables having independent components that perform transformations on data and process the input they receive where pipes serve as connectors for the stream of data being transformed. As a result, using those different styles together will lead to high maintainability and modularity (via Pipes-and-Filter), reliability and usability (via Event-Driven asynchronous events), and security (via Shared Memory's user management system on a shared database). That is, complex systems consist of many different components and functionalities that require using different architectural styles to achieve those set of functionalities, connect between those components, and limit the risks caused by one of the architectures.

However, the idea of representing systems as different architectures might be a slight downside for the understandability and readability attribute of the software by making it slower to understand the code of a specific component/feature because of how it could be linked to another component in a different architecture, so a well-detailed documentation must be written to avoid that. In addition, this also badly affects the maintainability because then fixes will not only have to be applied to a single component but also to all affected components in other architectures that rely on that component, which makes maintenance longer unless a well isolated components were implemented. Furthermore, using more than one architecture makes it harder to reuse that system's components because of how they might be connected to each other that it will be hard to decouple them and reuse in other systems, unless the other system to be reused in has identical requirements as the source system (like air versus marine traffic control systems). However, for logical errors, I believe using multiple architectures has no significant effect on their occurrence if they are as well written and detailed as if a single architecture was used. Similarly, I do not believe using multiple architectures has significant effect on schedule and budget outruns, if they were thoroughly analyzed and all high-impact factors were taken into consideration.

3. Tiered Architectures

3. A couple of the group design projects this semester used some version of a client-server architecture as the top-level organizational feature of the design. In *Just Enough Software Architecture*, this style is covered in Section 14.12: "Client-server Style and N-tier". From the perspective of the material about design and assessment presented in this class, answer the following questions about client-server architectures:

   a) [10 pts., 1 page limit] From McCall's list of software attributes defining quality, identify the five attributes you believe **most benefit** from using a client-server architecture. For each of the five, briefly describe **how** such an architecture improves the attribute.

      I believe the five McCall's attributes that most benefit from using a client-server architecture are: integrity, maintainability, flexibility, reusability, and portability. I believe the architecture greatly affects and improves integrity by adequately securing data because of the centralized design that only allows authorized users to access the data within login and password as well as two-factor authentication. In addition, if the data is lost, the records can be recovered quickly with one backup.

      Moreover, I also believe the client-server architecture improves maintainability because of how it isolates the client components from the server ones, so you will only have to worry about maintaining the faulted server/client components without affecting other client/server components. Besides, the centralized design of backup, security and antivirus also makes it easier to setup and troubleshoot because data backup and all other concerned elements are tackled centrally where everything takes place at one physical server. Besides, fewer support staff are needed to manage centralized security accounts than that would be needed if security and resource access had to be configured on each individual computer on the network.

      In addition, I think isolation between client and server components hugely supports the flexibility attribute by allowing making changes to client components without affecting other server components or dependencies and vice versa. Besides, the organization of the system as two different layers (client and server layers) makes it easy to understand and know what changes need to be made to what layer, which makes the modification process even faster.

      Furthermore, I also believe that the client-server isolation allows the system's components to be easily reused either within the same system or in a different system by splitting that system into multiple layers and components. For example, HTML, CSS, and JavaScript client files developed for a payment checkout feature for an online shopping system that is used for a cart's payment checkout feature can easily be reused in either the same system for a membership subscription's payment checkout feature or a completely different system like online learning system's membership payment checkout feature.

      Lastly, I believe that the client-server architecture improves the portability attribute because of how the client can be ported and still use the same server. Besides, clients also have the ability to share any resources at various platforms in addition to enabling the server to be replaced, restored, upgraded, and moved without influencing the client.

3. Tiered Architectures

b) [5 pts., ½ page limit] Identify the **critical features** of a design problem that make it a **good match** for a client-server solution, and **explain why**.

I believe the client-server solution is a good match for systems that require a separation or abstraction of concerns between the client and server, where it can improve performance in scalability, so that new features could easily be added to the system without affecting many components of the system.

In addition, I believe it is also a good match for systems that require separation of functionality where its layers, client and server could complete tasks independently in the sense that request and input validation are handled on the client side while processing client's request and returning results are handled on the server side e.g., the client does not need to know how the server handles user authentication or request validation and server does not need to know how client handles HTML inputs validation.

Lastly, I believe that systems that deal with multiple requests and users would also be a good fit for the client-server architecture because of the ability of each layer to function efficiently at large scale as a result of the separation of functionality that solves scalability challenges like load balancing and partitioning.

c) [5 pts., ½ page limit] Briefly describe a problem scenario where a client-server architecture would be a **poor choice** and identify **which aspects of the problem** are important in making this judgment.

I believe using a client-server architecture for a new startup that have a new videotelephony application idea (something like FaceTime or Zoom) would be a poor choice for various aspects like maintenance, single point of failure, and traffic congestion.

Firstly, the cost of configuring the server hardware components and software tools is very high that a new startup could not afford in addition to the high cost of hiring technical as well as skilled staff for maintenance of the application specifically for server machines.

Furthermore, since the client-server architecture fully relies on the central server, if it fails, all client/users' request cannot be done, that a real-time application cannot support such failure since it requires 100% reliable and active servers with almost 0% downtime.

Lastly, since the application completely relies on real-time data for audio or video, large number of simultaneous requests will be sent to the same server which might cause the server to slow down or even shut down causing the whole application to crash.

4. Sea Buoy Problem

4. [15 pts., ½ page limit]
One approach to measuring software quality was discussed in Lesson 10: the OPA Framework. In this framework, software quality indicators (SQI's) are measured. In describing the OPA Framework, Dr. Nance mentioned that scores for SQIs ranged from –5 to +5, and that there was little absolute meaning to this scale. Instead, one must have a history of applying the SQIs to several projects (or many times to one project) so that SQI scores can be compared against each other.

Suppose your organization wishes to adopt an OPA approach to software quality measurement, and they will begin with a project on which you are currently working. If this is the case, the first time SQI scores are calculated for your project, you will have no historical data to compare against, either from this project or from other projects. Explain how this first set of SQI scores can still be used to point out problems with the quality of your current project.

Although no historical data to compare against exists, I believe this first set of SQI scores can still be used to give us a rough measurement of quality attributes and point out problems within the project's individual components themselves. For example, if we give the readability quality of a well written and documented frontend part a score of +4, that means we can give a score -4 to a poorly written and documented backend part because then we have an idea of what a "+4" part should look like.

In addition, we can also use those scores to compare between the software quality attributes in a more meaningful and reasonable way instead of just directly measuring them. For example, if we gave the efficiency quality of the project a score of +5 and portability quality a score of +2, it would indicate that the software is perfectly efficient and needs no more efficiency enhancements, while the portability quality is fairly good but still needs some improvement to be considered "perfect". That is, those scores give the current project a more meaningful and reasonable way to evaluate and compare between its own qualities than just saying this software is efficient or portable.

Lastly, this first set of SQI scores can also be used as the base to be compared to for the next phase of the current project to measure the enhancements happened during such phase. For example, if a fair reliability quality has a score of +1 in this first set of scores and the next phase hugely enhanced it to be great, then we can give it +4 as the new score, since we already know that +1 is considered fair, so the new score must be way higher than that, hence it's +4.

4. Sea Buoy Problem

5. Review the Chef-It! design project (https://nyteknight.github.io/cs5744P1/).

a) [10 pts., ½ page limit]
Identify and describe the most significant weakness(es) you see in this design.

Since this design relies heavily on the microservices architecture, I believe one of the most significant weaknesses is the high degree of complexity because of how microservices are independent, that makes it requiring more development time and coordination than other architectures like monolithic would need, in addition to the difficulty of tracking down errors and resolving them.

Furthermore, since microservices are deployed independently, it will take longer time to get them all up and running, which increases the downtime of the application in case it crashed and needed a redeploy. In addition, the self-contained nature of microservices is another weakness because of how they rely heavily on the network to communicate with each other resulting in slower response time and increased network traffic that affects the user experience.

Besides, it will be difficult to track down errors caused by multiple microservices communicating with each other affecting badly the maintainability attribute of the system. Moreover, the nature of having the application spread out across multiple servers and devices makes it harder to test and debug because of the limited access to all servers and devices that are part of the system.

As a result, all of that will require a skilled development team having multiple technologies skillset to be able to implement, manage, and maintain the microservice-based design which adds additional costs.

b) [5 pts., ½ page limit]
Is this design likely to lead to a high-quality implementation? Justify your answer.

I believe this design will lead to a high-quality implementation from various aspects like flexibility, reusability, and integrity. For flexibility, I believe the independence of microservices hugely supports the flexibility attribute by allowing making changes to each component without affecting other services. Besides, the organization of the system as services makes it easy to understand and determine what changes need to be made to what service, which makes the modification process even faster and more efficient.

In addition, using microservices architecture allows the application services to be sharable across either the application itself or other systems, which enhances the reusability attribute of the system. For example, if the CHEF-IT organization has several different areas, each with a login or payment option, the same CHEF-IT microservice application's login or payment services can be used in each of those instances.

Lastly, I believe that breaking down the system's components into smaller pieces/microservices leads to protecting sensitive data from intrusions from another area. Besides, using secure APIs safeguards data by ensuring that only authorized users, applications, and services are allowed to make requests to those APIs. Accordingly, all of that lead to improving the integrity attribute of the system.

REFERENCES

Use the space below to include any bibliographic citations to other work to which you refer in your answers, of any.