Software Engineering

Department of Electrical Engineering & Electronics

December 1, 2021

Abstract

A good software engineer knows how to arrange software projects reasonably and complete projects efficiently with high quality as possible. In this report, a restaurant management system will be assumed with some requirements to learn the process of software system development cycle. There are three different roles in the restaurant system, which are Manager, chef and customer. Each role will have different functions which are also assumed. Moreover, ten questions about software engineering will be explained in detail. By reading this report, readers can clearly understand the process of completing a software project and some factors affecting software quality will also be introduced. Additionally, the future development of this system will also been discussed.

Declaration

I confirm that I have read and understood the University's definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with any other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences of engaging in plagiarism and collusion as described in the Code of Practice on Assessment.

Contents

1 Question 1

State the problem briefly; what software are you proposing and why?

My team is employed by a restaurant to design a system for the management of restaurant including three characters: Manager, Chef and Customer. For everyone in the team, the work will be distributed reasonably. The three characters required by the restaurant all have different functions: The manager can **view and change** all the information and **create** a new account. The chef needs to **check the raw materials** and **change the menu** in time. Customers need to **order and checkout** when preparing to leave the restaurant.

When designing this program, the C++ language is chosen and the compiler is the latest version of **Visual studio**. The reasons for the selection are as follows:

1. **Fast and Powerful**

   Since C++ is a compiler-based programming language which means that they are pre-interpreted, which makes the code faster and more powerful [1]. Therefore, the efficiency will be improved bu using the C++ language.

2. **Huge Community**

   C++ has a huge community to help us finish this problem comparing with other language[1].

3. **The convenience of class**

   Since various **classes** can be set in C++, the design of the program will be simpler for the problem with different characters. In addition, in terms of **confidentiality**, C++ can make confidential processing on the source code of the program.

4. **Cross-platform support**

   Since Visual studio can be used in Windows, Linux and Mac platforms, which may be more convenient for a teamwork.

2 Question 2

Write down the proposed product specification: an initial list of requirements (not yet analysed) that the customer wants and needs in the software to be developed. Include some functional, non-functional technical requirements.

Since the project involves three roles, all functions will also be carried out around the three roles. The most basic requirements have been mentioned above. In this part, the product specification will be carried out in more detail and the defects of this system will be evaluated. Moreover, different authority will be provided to different type of users. Therefore, the product specification designed is as follows:

1. **Manager specification**

   (a) Eligible to **create and browse** accounts, including chef and customer accounts.

   (b) Qualified to **change** all information, including all input or output. Manager has the **greatest authority**.

2. **Chef specification**

(a) The chef can **view** the food information stored in the warehouse and make changes to the menu(If a certain dish does not have enough raw materials in the warehouse, the dish needs to be deleted from the menu).

(b) The chef can **increase or decrease** the types of dishes on the menu.

(c) When the chef adds dishes, **duplicate dishes** can not be added.

3. **Customer specification**

   (a) Customers need to be able to **view** the menu and **choose** their own food from the menu

   (b) Customers need to pay on this system after the meal is over

Chefs, consumers and managers can use the restaurant management system individually , which can be more efficient management of the restaurant.

Restrictions on this system: Since only Visual studio is used for compilation, the interaction between the user and the system is only to select or change numbers and words, which is relatively **monotonous**. The system will be hard to attract users' interest and users who use this software for a long time may feel bored and **reduce work efficiency**. The next development is to use colorful **web pages or mobile application** to reproduce this system, which can better attract users. In the future, the methods of using multiple software may be also considered to increase this possibility.

3 Question 3

Do a preliminary requirements analysis: analyse the initial requirements you listed in Question 2, discussing e.g. potential conflicts or interactions between the requirements and considering development constraints, resources, risks, etc... You may wish to use a MoSCoW list as part of the analysis. (If relevant, note down also any changes to initial requirements from this analysis.)

Several different analysis perspectives will be used in this section, and the MoSCoW (Must, Should, Could, Would) will also be used in the analysis.

1. **Risk analysis**

   Since this part involves three roles and each role has different rights, which means that the system **should** be designed to ensure that each user can only use his own interface and cannot use the functions of other roles. For example, if the program design is wrong (customers can use the manager functions), which **could** cause great losses to the company's interests. This means that after completing the procedures, our team **must** also be responsible for follow-up **training**. Once users encounter problems, the solutions **should** be provided in time.

2. **Feasibility analysis**

   From the perspective of program design, different considerations **must** be required for each characters' interface. In addition, the login interface **would** be designed and the interface will distinguish different roles. Meanwhile, managers also need to create new users accounts and other operations on this interface. Considering that there are three roles in this system, but only two classes **could** be designed: chef and Customer. For

manager, manager **could** log in to their interface to **complete their operations**, which means that the manager has the highest authority.

3. **Difficulty and workload analysis**

   The difficulty of designing this system lies in the design of the main interface: the interface of the chef, customer and the interaction between the manager and chef, customer (The manager **must** be able to log in to the interface of the chef and the customer, and vice versa). There are exactly 5 people in our team, which means I (As the team leader) **should** let them choose what they are responsible for. For me, I **would** be responsible for the connection and coordination of each team member. The workload of each job **should** be approximately the same, and the work can be carried out more fairly.

4. **Potential conflicts**

   Potential risks **could** arise from program design. Once a certain team member encounters a problem, I **must** support him in time and work with him to find a solution. In addition, since this is a commercial software, the related intellectual property authorization work **must** be completed.

After the above analysis, changes **could** be made to some of the functions of the Manager. Since only the manager is required to select the interface of other roles, and there is no need to redesign an interface specifically customized for the manager. Therefore, the specification of the manager **could** be changed to "Qualified to change all information, including all input or output.". Manager **would** choose to enter the interface of the chef and customer to complete their work, which can reduce the workload and achieve the same function.

4 Question 4

List some key deliverables of the project (e.g. user documentation. . .)

The deliverables describe the objects that must be completed on a specified due date. Deliverables can be reports or services-tangible or intangible [2, 3]. The company needs to provide users with the following key deliverables:

1. **Software product**

   Obviously, the final software needs to be handed over to the user. If the user is unsatisfied with some functions, the program needs to be adjusted in time.

2. **Testing results**

   Before handing over the program to the user, the necessary tests need to be done in advance. The results of this part also need to be presented to the user.

3. **Source code**

   The submission of source code requires careful consideration. The source code needs to be treated confidentially when necessary. Choosing c++ and compiler: Visual studio also has this advantage because they may protect the source code.

4. **User documentation**

   User manual (documentation) needs to be provided to users together with product to help users understand the operating rules of the program, and it is also convenient for users to use this system.

5. **Prototype**

   At the same time, the prototype of program also needs to be provided to users, because it can be complete iterative development of an application [2].

6. **Reports**

   The report can include product introduction, product development process.

7. **Possible mobile application**

   Considering that this product may be more convenient to complete on a mobile phone, it is necessary to design a mobile phone application in the future.

8. **Users training and contact information**

   For safety reasons, users may need to be trained in the future to prevent users from using the program incorrectly and causing economic losses. In addition, once users encounter difficulties that need to be solved, it is also necessary to provide the necessary contact information of engineers.

5 Question 5

Identify the software development process to be followed and why you have chosen it for developing this product. (E.g. waterfall model, spiral model, etc. – or, more likely, some combination of approaches.)

In this part, the Waterfall model will be used to clarify the software development process. For V model, since the sequential phases will be needed to be functional, the cost will be higher than Waterfall model and the complexity is also more than waterfall model. On the other hand, due to the fact that Waterfall model will use linear development and only one phase is operational, the cost and the complexity will be low than other models (V model) [4]. The following is the waterfall model designed and each process will be explained.

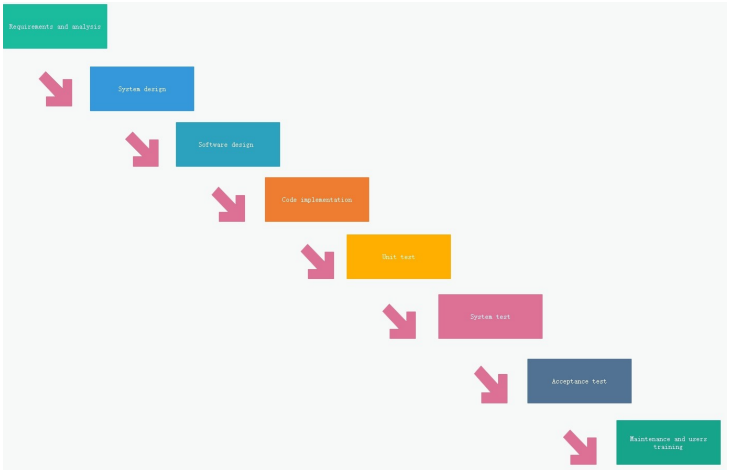

Figure 1: Waterfall model

1. **Requirements and analysis**

   This part mainly analyzes the needs of users and summarizes important functions. Requirements have already been answered in the previous question

2. **System design**

   This part is based on the above analysis and design of the system framework for the requirements. For example, our team need to think about how many roles are involved in this problem and the connection between each role.

3. **Software design**

   This part includes the classes that need to be created. In this question, we need to create a chef class, customer class. In addition, we also need to think about how the system works and how many several interfaces are required to complete.

4. **Code implementation**

   This part needs to split the code into various parts, and analyze each functions.

5. **Unit test**

   This part mainly involves: Conformance to specification, processing accuracy, checking whether the running logic of the program conforms to the specification and determining the degree of requirements realization [2]. For the restaurant system designed, a total of three roles have to be tested about whether the function works normally.

6. **System test**

   It is necessary to test whether the designed system meets the requirements of users, whether the interactive interface of the three roles is loaded normally and test whether all functions and operations for this system are useful and successfully completed [2]. More importantly, test whether all modules are compatible in the system and whether there are functions that can be restored when the program has problems.

7. **Acceptance test**

   Test whether the designed program can be used normally by the users without violating any laws and regulations.

8. **Maintenance and users training**

   After the product is delivered to the user, later maintenance is also necessary, which specifically includes: corrective maintenance, perfect maintenance, adaptive maintenance, preventative maintenance [2]. The specific types above will be explained in Question 10. In addition, it is necessary to train the users after completing this project.

6 Question 6

Write an overall plan, showing the main tasks, and the activities, milestones and deliverables associated with each task. Your plan should focus in on software design principles appropriate to your product, and also, refer to the software development process chosen in Question 5. Your plan must clearly incorporate the relevance of abstraction and modularity.

This part will involve an overall plan. The specific plan will be designed from the development process and abstraction and modularity. The abstraction and modularity of the plan are as follows:

1. **Abstraction**

   Abstraction is important to program structurally [2], which is divided into three levels: Top, second and lower levels. Therefore, the **plan** designed for each level is as follows:

   (a) Top level

   The main problems and specifications will be planned at this level. For this system, there are three types of characters using the program, and each has a corresponding interactive interface. For the manager, manager can **create, read, and change** all the information of chefs and customers. For the chef, the chef's interface needs to **read** the chef's input and **display** the changed output. For customers, customers need to **view** the menu information and make choices, and finally program needs to **calculate** the customer's consumption and display it.

   (b) Second level

   The major tasks will be identified and a list about it will be needed. Hence, the list is as follows:

   i. Managers **create** an account,**view and change** all the information entered by chefs or customers
   ii. Chef needs to **view and change** the inventory, and chef also has the ability to change the information on the menu
   iii. Customer needs to **view** the menu and make a selection (Input), and finally **checkout**

   (c) Lower level

   More details will be provided in this level.

   For Manager:

   i. Open a file
   ii. Create an account, including password, user name, and the role: chef or customer.
   iii. Design options to allow the manager to choose whose interface to view or change information (Chef and customer)

   For Chef

   i. Enter the user name and password to enter the interface
   ii. Check the stock in the inventory
   iii. Change the specific amount of raw materials
   iv. View or change menu

   For customer

   i. View menu
   ii. Choose dishes
   iii. View selected dishes
   iv. Checkout (calculate the total meal cost)

2. **Modularity**

   A system has a number of modules to complete the whole tasks. Each module will only complete simple task [2] separately and each module should have strong cohesion, weak coupling and easy to change. The specific modularity will be divided into several parts and introduced separately.

In terms of **completeness**, the designed system should ensure that all the requirements of the user are covered. In terms of **compatiblity**, the designed system needs to be tested separately on the windows, Linux and Ios system to ensure that it can run normally on all common platforms. In terms of **portability**, we consider setting an application in the mobile phone in the future. It is much more convenient to use the application in the mobile phone than to view it on the computer. In terms of **operating speed**, we pay attention to the improvement of loops without using too many worthless loops. For these useless loops, we need to find a suitable method to replace them. In terms of **communication**, all team members should integrate their code and upload it to a platform that all team members can view, such as **Darcs**. Therefore, if there is a problem, other team members can view and make changes at any time. In the interface design, the design should be as simple as possible and easy to understand, so that the logic presented to the user will be not particularly complicated.

3. **Other plans including test and maintenance**

   According to the introduction in Question 5, the unit system acceptance test will be carried out. Therefore, all required functions of the user will be tested to ensure that all functions meet the requirements. The normal connection between each module will also be tested to ensure that the system is working properly. Cross-platform testing will also be conducted to ensure that the system can be used on common platforms. After the product is delivered to the user, follow-up training will also be carried out simultaneously. Suggestions from the users will also be taken. Once users have any problems that need to be solved, a dedicated engineer will be required to solve them.

7 Question 7

Suggest and briefly justify an appropriate version/revision control strategy for the product's source code, to allow your team of 5 developers to collaborate while tracking / controlling changes.

In the software development process, **revision control** is the management of changes made over time. These can be source code, project contents, or any other information related to the project. It allows multiple people to work on the same part of the project without worrying about their changes overwriting the work of others [5]. The revision control system is usually hosted on a networked server.

The specific steps are as follows: If the group member creates a new file that should be part of the project, the file must be added to the **repository**. After uploading the file to the repository, anyone else involved in the project can view and use the file. If the group member would like to edit a file that is already part of the project, the file must be checked out [5]. If two member upload the similar files (Repeat), the system will announce a conflict and let users check the difference.

After checking the information, the platform I choose is **Darcs**. Darcs is a revision control system released under the GNU and GPL. It supports many platforms including Linux, Windows, MacOS X, FreeBSD, OpenBSD, NetBSD, DragonFlyBSD, Solaris and AIX. It also comes with a CGI script that allows users to browse the source code repository [6] via the Web. Because Darcs supports editing on **multiple platforms**, it is more convenient for team members to operate. In addition, Darcs supports **offline mode**. This means that our own working directory is also a repository. Even if we can't access the server or have a poor network

connection, we can simply record our work. When the connection is good, we can execute **Darcs push** to transmit it to the public server. In addition, Darcs can easily collaborate **via email**. If we want to add features or bug fixes to a project, we can make a local clone. After applying any changes, then we can send the patch via email (darcs sent) [6].

8 Question 8

Discuss various measures of quality, in specific relation to your software.

Measuring the quality of software depends on the following aspects, which are: Software methods, Standards and procedures, formal technical, reviews software configuration and management Test [2].

In addition, the factors that can determine the quality includes reliability, Efficiency, Integrity, Usability, Testability and maintainability. After the product is completed, it must be **verified and validated**. The following will discuss software quality in all aspects according to each different part. When analyzing different software assurance, it will be specifically combined with the factors that affect quality.

1. **Software methods**

   In this project, Visual studio was chosen as the compiler of the program. Visual studio can edit the program more conveniently and also easier to share. Moreover, using Visual studio can improve the **efficiency** of the program. In addition, the project can also be divided into multiple modules by designing the classes, and improving the connection between each part can also improve the **integrity** of the system.

2. **Standards and procedures**

   After each team member completes his work, each of them should upload his work to **Darcs** (Revision control platform), which will convenient for other team members to view and change. In addition, the work can be completed when there is no network connection. And then, upload the work to the Darcs when there is connection, which can also be completed with more **flexibility**.

3. **Formal technical reviews**

   After each team member completes his work, for example, after completing the interface of each role. The next step is to assemble the parts they have completed into one system. At this time, we need to have a meeting in our group to discuss **technical reviews** to determine the next step of the assembly process. After the review, the errors in the process of work can be reduced as much as possible, and the **correctness** can be improved.

4. **Test**

   After completing this system, three different interfaces' tests also need to be completed. For example, test whether the manager can create an account and change information. If the system accurately completes all requirements, then the program has **testability**. In addition, the connection between each interface also needs to be considered. It is necessary to test whether the user can normally "interact" with the program. A good degree of interaction indicates that the system has **interoprtability**.

5. **Software maintenance management**

   After the software is handed over to the user, post-maintenance is also very necessary. Customer needs to use the designed system to test whether the system meets the requirements. If users encounter any problems and give us feedback, we can also change this product in time to improve **maintainability**.

Through the design method of dividing the program into modules, the requirements can be completed to the greatest extent. The process of designing software determines the quality of the final product. Good software quality will directly depend on the above factors.

9 Question 9

Suggest and justify either prototyping, or testing/validation, strategies which can be shown to be appropriate for your product

White box testing strategy will be adopted, which can verify an application while its internal structure, design, and implementation details are available to testers [7]. Compared with other testing strategies, white box pays more attention to the logic relationship between systems. The main level of white box testing works by focusing on multiple units present in the application [7]. In white box testing, the tester will select the input to go through the code execution path and parse the appropriate output.

For the restaurant system, the white box test is very necessary because we can find the cause of the problem from the source code when a problem occurs. Although the white box test requires a high degree of understanding of the program, the white box still plays an important role in this system.

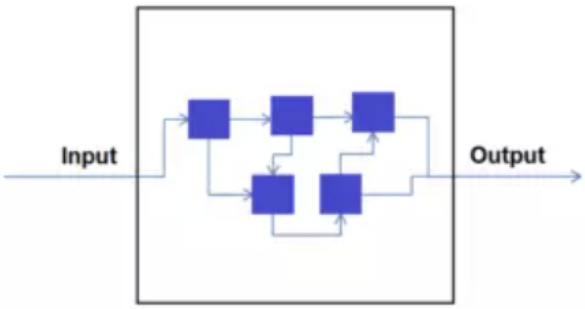

Figure 2: White Box Testing (taken from [7])

For example, the white box test strategy is needed when the checkout function can be realized. Since the system need to multiply the amount of all the dishes by the quantity when customers check out, which involves mathematical operations. Therefore, the white box test can help with mathematical proofs to verify that whether the program **calculate the correct amount**. Additionally, when the manager creates an account, a white box test will also be needed to verify whether the function of creating an account is correctly implemented.

In addition, since the restaurant system involves multiple modules, the **Top down testing strategy** is also used to verify the working conditions of each module. Top-down testing is an

integration testing technique used to simulate the behavior of low-level modules that have not yet been integrated [8, 2].

For the restaurant system, there are a total of three interfaces to be designed. If we treat each interface as a module, we can better solve problems by testing it separately. For example, testing the function of chef can be started when the chef module is completed, which can improve the efficiency and quality of the project. On the contrary, it is difficult to find the problems if testing after integrating the three modules. Finding the problem and solving it before the integration can better complete the requirements. In this case, the quality of the software should be improved through the iterative nature of the test [2].

10 Question 10

Briefly discuss the scope for further developments, enhancements etc. as part of the anticipated software maintenance after delivery.

When discussing the maintenance of a product, three maintenance perspectives need to be considered: corrective, perfective and adaptive maintenance. The maintenance of the product is also a very important part of the product. The specific maintenance of the product is as follows:

1. **Corrective maintenance**

   In this maintenance aspect, fixing bugs is the main function. Bugs caused by loops or other errors in the code need to be solved in this part.

2. **Perfective maintenance**

   After the restaurant system is handed over to the user, the user may put forward some opinions about the program. Improving procedures based on these opinions is also a kind of maintenance. For example, the customers may feel that the ordering process is too complicated, which means that we need to optimize this part of the code according to the user's requirements in the next steps.

3. **Adaptive maintenance**

   Since this system involves payment functions, relevant laws and regulations which also need to be paid special attention. Once these financial regulations are changed, the system also needs to be updated in time to match these financial regulations. For example, there may be no mobile payment like Apple pay in the original payment method, and new payment methods should also be added in time to adapt to the changing environment. In addition, users may find it inconvenient for operating this system in the computer. Therefore, the next step may be porting this system to the mobile phone.

4. **Preventative maintenance**

   It is very common for restaurant systems to change dishes and inventory. Therefore, if we want to optimize the code, we can consider deleting and changing dishes **in batches**. In addition, improving the **stability** of the system is also an aspect to be considered, and the designed product should be able to operate stably for a relatively long period of time.

In addition, due to the particularity of the restaurant system, the training of using this system is also an essential stage to reduce the loss of the interests caused by improper behavior. For future development, this system should be transplanted to mobile phones instead. Completing

work through a mobile application can increase the **efficiency** of the work to a greater extent because a computer operating system still has certain limitations, such as hard to carry, the low speed of changing information. Therefore, it is very necessary to develop a mobile application.

References

[1] TechVidvan, "Advantages and disadvantages of c++," https://techvidvan.com/tutorials/cpp-pros-and-cons, 2021.

[2] D. G. McIntosh, "Software engineering," https://liverpool.instructure.com/courses/46000/pages/software-engineering?module_item_id=1259000, Department of Electrical Engineering Electronics, 2021.

[3] Corporate finance institute, "Deliverables," https://corporatefinanceinstitute.com/resources/knowledge/other/deliverables, Springer, 2021.

[4] N.Sharma, "Difference between v-model and waterfall model," https://www.tutorialspoint.com/difference-between-v-model-and-waterfall-model, 2020.

[5] Computer Hope, "Revision control," https://www.computerhope.com/jargon/r/revision-control.htm, 2021.

[6] The free country, "Free source code version control management software," https://www.thefreecountry.com/programming/versioncontrol.shtml, 2021.

[7] M.Agarwal, "White box testing – a practical guide for all beginners," https://www.techbeamers.com/white-box-testing, 2021.

[8] Tutorials point, "Top down integration testing," https://www.tutorialspoint.com/software_testing_dictionary/top_down_integration_testing.htm, 2021.