UNIT 13 SOFTWARE TESTING

Assignment 1

Learning Aim A

Understand the software development and testing methodologies commonly used during the development life cycle to quality assure software.

Oliver Collins-Cope
2102775@rutc.ac.uk

Contents

Introduction

Throughout this unit I will explore the importance of software testing and development process as well as different types of testing the tools and techniques used to perform them. In this paper I will be focusing on the fundamental principles of software testing. This paper will aim to document the various types of testing, their significance in the software development life cycle, and the importance of using different testing tools and techniques in order to ensure that the quality of the final product is maintained. Finally, this paper will also document use cases of different testing methodologies.

User requirements and typical software job roles

User requirements

The user requirements of the specific needs, wants, and expectations of the stakeholders who will be using the software system. Understanding these requirements, such as through gathering or defining them, is a vital step in the software development life cycle. This is due to the fact that it ensures the software system is designed undeveloped with its initial intended audience at the forefront of development.

From the perspective of a software tester, it is crucial to have an in depth understanding over the user requirements. This will ensure that when software testing, the software tester will be able to ensure that the software is meeting the user requirements. Furthermore an in depth understanding of the user requirements would allow the software tester to review and analyse them, giving them the opportunity to identify any potential issues or conflicts and raising these issues with the development team.

Finally an in depth understanding of user requirements will aid in identifying the scope of testing, such as how complex or critical the testing should be of the software, and therefore help in developing appropriate testing strategies for the software. Through the identification of different parts of the software such as features, functions, or other relevant aspects the end users might interact with, the software tester can develop different test cases that will aid in the applicable use cases and scenarios.

Typical job roles

Software developers

One typical job role in a software development team is a software developer. Among others, some of the job responsibilities of a software developer include:

- Developing software applications, this is where software developers design, develop, and maintain software through the use of different tools such as programming languages and other available development tools.
- Write and maintain code, this is why software developers writing and maintain code from the software applications that will later be used in testing.

Some typical skills of a software developer include:

- Programming skills
- Problem solving skills

- Communication skills
- Attention to detail
- Time management skills
- Adaptability

A combination of all of these things allow software developers to play a vital role in the software development life cycle, including job duties like designing, developing, and maintaining the software applications that they create.

Testers

Software testers play a crucial role in the software development life cycle, as they are responsible for ensuring the software applications meet the end user requirements. Furthermore software testers are responsible for testing a variety of aspects of software including, functionality, usability, performance, and security. Some job responsibilities include:

- Developing and executing test plans, software testers create test plans to ensure that the applications they are testing successfully meet the end user requirements.
- Identifying and communicating bugs, software testers focus on finding issues and bugs within the software during their testing periods and work with the software developers to ensure that the bugs are resolved before the final release.
- Analysing test results, software testers must analyse the results of the software testing in order to ensure that any issues are found and consequently reported

Some typical skills of a tester include:

- Knowledge of software testing methodologies
- Attention to detail
- Communication skills
- Technical ability

Business analysts

Contrary to software test is a software developers, business analysts play crucial role in ensuring in software applications align with the overall business strategy as outlined previously in the development of the project. Typically they have a large skill set ranging from business domain knowledge, analytical skills, and communication skills. This enables them to effectively complete their job as business analysts. Some job responsibilities include:

- Gather and analyse requirements, due to the nature as analysts they must analyse very acquirements from stakeholders, such as the users of the product and customers, in order to be able to translate their needs into functional requirements that the software developers and testers can use to create a successful product.
- Documentation, as business analysts are required To document all of the functional and non-functional requirements they have successfully analysed.

Some typical skills of a business analyst include:

- Business domain knowledge
- Analytical and problem-solving skills
- Communication skills
- Technical ability

Project managers

Project managers play more of a central role in the development of software. Rather than having one specified function, they are often required to branch out and perform a wide range of functions in order to ensure a successful project. Some job responsibilities include:

- Defining the project scope, this is why project managers identify the goals and objectives of the project and through this develop an acceptable timeline.
- Develop project plans, based on the previously identified goals, objectives, and timeline, project managers develop project plans that outline the tasks that must be completed within the projects time.
- Managing project teams, this is where project managers manage the teams by assigning task and monitoring their progress.
- Monitoring and controlling projects risks, project managers help to identify and monitor any project risks that arise through the development of risk mitigation strategies, and taking corrective action when is necessary for the project.

Some typical skills of a project manager include:

- Leadership skills
- Communication skills
- Time management skills
- Risk management skills

Product owners

Product owners require a combination of many skills including, customer focus, strategic thinking, communication, and analytical skills. These skills enable product owners to effectively play a crucial role in defining product vision and strategy. Some job responsibilities include:

- Defining product vision and strategy, this is where product owners have to define the product vision and strategy through the identification of customer needs which is typically provided by the business analyst.
- Developing product road maps, this is where product owners develop road maps that outline the future product features and functionality including timelines and delivery schedules in order to present a base timeline for the project.
- Prioritising product backlog, this is where product owners have to prioritise different backlog of features based on which features and functionality is most important valuable to the customer.
- Testing and validating product features, finally product owners have to test and validate product features to ensure that they meet the requirements of the customer that they previously outlined for the development team.

Some typical skills in a product owner include:

- Customer focus
- Strategic thinking
- Communication skills
- Analytical and problem-solving skills

Characteristics of common software testing methodologies

Unit testing

Unit testing is a software testing technique that involves testing individual parts of the software application while it is isolated. Further elaborate on this, this means that unit testing involves testing each individual part of the code separately in order to ensure that its functions as intended and producing the expected output.

Some of the key features of unit testing include:

- It can be automated based on specialised testing tools or framework
- It is typically performed by developers and testers who have knowledge of the programming language it used to develop the application
- It usually involves test cases that are for specific situations
- Unit tests are designed to be able to test even the smallest parts of an application and this enables developers and testers to catch errors and defects early in the development life cycle
- Unit testing is especially useful in identifying and isolating individual units of code the hold bugs

Furthermore, unit testing is a kind of white box testing which means that the tester has access to the application code and internal workings. It is important to make this differentiation as black box testing, such as system or acceptance testing, focuses purely on the external behaviour of an application and therefore does not access the code of the software. (TechTarget Contributor, 2023)

Acceptance testing

Acceptance testing is a software testing technique that focuses more on validating a software based on specific requirements that are previously outline and detailed in order to be accurately used. This means that unlike unit testing, it does not look at the code.

Some of the key features of acceptance testing include:

- Acceptance testing is performed on the complete software application to ensure that the final product meets the specific requirements of the users
- Acceptance testing focuses on validating the applications based on requirements, rather than catching errors and bugs
- Contrary to unit testing, acceptance testing is completed by testers who are meant to emulate end users, or the application stakeholders as opposed to developers
- Typically, acceptance testing needs manual testing which is different compared to unit testing

Finally, acceptance testing is a kind of black box testing, meaning that the tester does not have access to the code of the application and therefore only works with the developed software. It has a completely different objective compared to something like unit testing and does not occur until near the end of the software development life cycle. (Gillis, 2023)

Functional testing

Functional testing is a type of testing that focuses on verifying the functionality of a software application. It works via testing multiple different functions and features, ensuring that the capabilities of the specified requirements.

Some of the key aspects of functional testing include:

- As the name implies, functional testing focuses on testing the functionality and verifying that all of the features of the program are working as intended. This can include testing various things like inputs and outputs, and how the application performs under different situations.
- Functionality testing also aims to test the requirements of the software and validate that the application meets the requirements of the specifications.
- In functional testing, you usually test things like mainline functions of the applications, basic usability that dictates whether or not the user can freely use the applications, accessibility testing, and error conditions where you check if errors are reacting in the way they should be.

Functional testing is also a kind of white box testing, allowing testers access to the source code while testing. This enables them to make sure that all aspects of the program are working as intended and therefore accepted as a final product. Functional testing can be performed manually or made to be automated, allowing for testers and developers to create a successful product. (Microfocus, 2023)

System testing

System testing is a type of software testing that is aimed at verifying the functionality of a software application in its entirety. It involves testing the various functions, features, and capabilities of the application as a whole, to ensure that it works as intended and meets the specified requirements. The goal of system testing is to ensure that the application works as intended and meets the specified requirements, including a series of features and processes that ensure its functionality and performance align with client expectations.

Some of the key aspects of system testing include:

- System testing involves verifying that all of the functions and features of the developed application are working and functioning correctly, even when integrated together at the end.
- Furthermore, it also aims to verify and validate that the software application meets the requirements from the document specification. These test cases are designed based on the requirements and therefore ensure that the tests are correctly used on the system.

- System testing also helps to identify defects and bugs in the software application, allowing for additional bug testing to ensure that the final product of the project is not defective due to these bugs.
- System testing can be performed either manually or it can be automated to allow for extensive testing without tester/developer downtime during this testing.

System testing can be considered both kinds of testing, both white box and black box, as it can be performed with the internal workings of the software available, alongside black box testing where it is performed simply with the software/application and no further knowledge of the source code. This means that system testing is a variable kind of testing and therefore helps to allow developers and testers, tests that can help to ensure it meets the clients' final expectations. (Tech Target, 2023)

Performance testing

Performance testing focuses on evaluating the ability and performance of the software/application. This helps to figure out the scalability of the application alongside the responsiveness, stability, and other application performance parameters.

Some of the key aspects of performance testing include:

- Testing under load, this is where the application is tested under numerous loads that impact the performance, such as user traffic, large loading assets/times. The goal of this is to assess how effective the application is at handling the different loads.
- Evaluating response time, this is where the tester checks how long it takes for the application to respond to various operations, such as loading, processing, multiple user bases, etc.
- Performance testing also aims to identify performance issues which the development team can aim to improve.
- Finally, performance testing also aims to see how the website handles stress, and what can be optimised in the software application.

Performance testing is both kinds of testing which means that the code is both not accessible, and accessible while performing this test. It largely depends on the scenario involved and the testing outcomes and can be modified based on what is needed of the test.

Security testing

Security testing, as the name implies, focuses on evaluating and measuring the security of a software application or system. This, similarly to performance testing, allows for both white box and black box testing. These tests are done under various conditions in order to identify as many vulnerabilities as possible.

Some of the key aspects of security testing include:

- The main aspect of security testing is to identify security vulnerabilities. This involves assessing the application for potential security vulnerabilities, such as authentication measures.
- Some security testing methods include penetration testing, vulnerability scanning, and compliance testing.

- A security configuration review is often performed in order to review the applications security configurations, such as firewall settings and encryption methods.
- Threat modelling is also performed where potential threats, vulnerabilities and weaknesses are identified in order to ensure they can be dealt with adequately.

Finally, security testing is used to identify and mitigate any potential security risks throughout the development process, so that they can be dealt with swiftly and resolved without risk of them being exposed and exploited. It is a crucial part of application testing as it aids greatly in protecting the applications and its data from unauthorised access and other security threat, protecting the developers and users from numerous troubles.

Regression testing

Regression testing is type of testing that focuses on evaluating the differences in the functionality of a software before and after updates, focusing on ensuring that the software remains functional, and nothing has deteriorated after the changes. It is primarily retesting the application using previous testing methods, in order to prevent defects introduced by changes or updates.

Some key aspects of regression testing include:

- Retesting previously tested functions, this enables retesting the application in order to measure the differences in the applications after updates.
- Identifying and fixing regressions, this is where any errors introduced through updates, aka regressions, and fixing them to prevent the application breaking.
- Automated regression testing is also a possibility, as this can be easily set up to check any differences in the application before and after changes.
- Regression testing can also involve prioritising test cases depending on the importance of critical updates as they might involve scenarios that take high priority and therefore have to be done as soon as possible.

Finally, regression testing can be done with both white box and black box testing methods, however it is commonly done with black box testing as the priority of regression testing is to identify if the application functions the same before and after the changes and updates, therefore making black box testing more common. (Katalon, 2023)

Stress testing

Stress testing is a type of testing that is similar to performance testing. It focuses primarily on identifying the performance of the software under extreme or stressful conditions, evaluating it based on predetermined requirements and specifications. It pushes an application beyond the normal operation capacity, this being the primary difference between stress and performance testing, in order to identify the weaknesses and strengths of the software and improve on both respectively.

Some of the key aspects include:

- Testing beyond the normal load, as stated above, stress testing focuses on pushing the application beyond the normal capacity of the software and this can be achieved

through several methods, such as higher loads, larger data sets, and increased user amounts.
- Stress testing also measures the performance of the application when it is experiencing degradation as this helps to outline future areas for improvements that do not behave, or function as well compared to others.
- Stress testing is also used to measure recovery and stability of the program, looking at the impact of the testing and how the application moves to recover the damage it suffered during the testing. This aids in giving the developers a deeper understanding of their software.
- Additionally, stress testing is used to identify the specific points of the application failing, called failure points. This allows developers to focus on these areas in the future for improvement.

Finally, stress testing is a type of black box testing, as the focus on the test is how the application responds, and therefore access to the source code is not required. It is essential to ensuring that the software can withstand extreme conditions and unexpected stress, while still performing adequately and optimally in those conditions. It also helps to identify any security vulnerabilities that might be identified during this testing.

Usability testing

Usability testing is a type of testing that focuses on evaluating the usability and user friendliness of the application from the perspective of the target/end user. This includes evaluating aspects like UI, accessibility, and many more features of the software. It often includes real users rather than simply testers, and some examples of large-scale testing is open betas where the users are able to access the software and provide feedback for a brief period of time to allow for improvements prior to the final product being released.

Some of the key aspects of usability testing includes:

- Testing with real users, as mentioned above, usability testing uses real users and this makes it unique amongst most testing methods, as some , like security testing, would never be able to incorporate end users in the testing process.
- Evaluation of user interactions, as the name suggests, it is crucial to evaluate the user experience with the software during usability testing, as this part of the testing if completely focused on how the software and end user are able to interact together.
- Assessing learnability, this is where usability testing forces on evaluation the learnability and intuitiveness of the application, such as how quickly users are able to understand and navigate the software. An example of this would be how quickly users are able to subconsciously navigate the software.
- User feedback and satisfaction, usability testing, given that it is focused on end user testing, requires user feedback and satisfaction measuring in order for the developers to gauge what needs to be improved on in the application and what should remain the same.

Given that usability testing often involves end users, it is commonly classified as black box testing. This is due to the fact that it is often not ideal for the developers to hand out the source code of their application given that this will likely lead to users exploiting the software and breaking it, making usability testing black box. It is essential to ensure that the software is usable and user-friendly, and this testing phase can help identify any issues that conflict with these goals, and therefore ensuring a positive user experience. (Usability gov, 2023)

Features of testing for different software development methodologies

Agile testing methodology

Within the Agile development methodology, testing is something that plays a crucial role as it remains an iterative process that happens constantly through the development of the software and is therefore crucial. The approach and features of testing in Agile includes:

- Software requirements specification, this is where the testing process starts. It depends on the understanding of the requirements specification, including outlining both functional and non-functional requirements that are translated from business needs. Additionally, user interface mock ups can be created that will aid in visualising the expected user functionality of the software.
- Test scripts and test cases, this is where the test scripts and cases are developed and created based on user stories created within the Agile methodology. These stories help to define ad test the user functionality in the software and therefore the scripts and test cases help to serve as a guideline for the testing team.
- Advantages of Agile testing, the agile development methodology offers many advantages for testing, including increased flexibility for changes that may come about as a result of testing, encouraging stakeholder involvement in the project, constant communication between the team, and improvements based on the reviews of the project. This focus on flexibility and adaptability is what promotes Agile as a strong development methodology.
- Disadvantages of agile testing, naturally, agile also has some disadvantages which are mainly the inability to estimate delivery times due to the complexity of the work, and the need for experienced developers and team members for Agile. Additionally, the focus on short term work can lead to aspects of the project being convoluted and more complicated than necessary. It requires careful and dedicated planning to be executed effectively.
- Agile methodology and changing requirements, due to the nature of agile development, it is well suited for changes. This leads to it being used in scenarios like unclear scope from clients and evolving products that change through the development life cycle. The iterative development of Agile allows for frequent external input and therefore helps to create a project that aims to satisfy the client through change.

Waterfall testing methodology

Unlike the Agile methodology, testing within the Waterfall methodology only begins right at the end of the development of the product. This means that the tests are not completed concurrently with the software development and often the effect of the development work is minimal on testing. The approach and features of testing in Waterfall include:

- Advantages of testing in Waterfall, the advantage of testing includes easy planning forwards and backwards, alongside a visible output at the end of the process. This approach allows for a linear development plan and provides a base product that the development team can focus on moving forward.
- Disadvantages of testing in Waterfall, the largest disadvantage of testing in Waterfall is by far the lack of flexibility in development. This means that any changes that might arise during the development process cannot be catered to due to the fact that any changes often lead to impacting the many resources, i.e., time, and money. Any unexpected risks or constraints that are not accommodated for might lead to the derailment of the entire project and issues that are discovered during testing may require significant reworking of the software, making this the complete opposite of Agile.
- Waterfall methodology and requirements, this development methodology is often used when the clients' requirements are well and clearly define, alongside previously completed projects. Additionally, a deep understanding of the technology involved in creation is often needed and there is limited expectation of any changes to be expected.

Kanban testing methodology

Contrary to both Agile and Waterfall, which usually have some kind of outline for when testing might occur, kanban introduces testing typically after the development of a feature, both in post-development and pre-production phases. The work in kanban is typically completed "just-in-time" and therefore the testing must reflect this ideology. To expand on this, that means that testing is usually performed per feature and expansion on the project, rather than as a whole, and is built on bit by bit. Finally, regression testing is a must for kanban, and should be performed prior to the complete products delivery. The approach and features of testing in kanban include:

- Advantages of testing in kanban, a large advantage of kanban testing is the continuous delivery of work and completed testing during the development periods of the project. Similarly to Agile, kanban allows much flexibility, often more so than Agile, and is much freer compared to Waterfall testing. This makes it much more suited for projects where the work is imminent, and changes are made based on evolving requirements.
- Disadvantages of testing in kanban, the largest downside for kanban testing is the lack of work leading to downtime for the developers and other team members. When there is no work flow for the team to focus on, there will be much downtime and this makes kanban unsuited for certain kinds of project that involve lots of periods where development does not take place, such as research projects. Additionally, projects that involve lots of planning do not mould together well with kanban's flexibility, therefore making it unsuitable for these projects.

- Kanban methodology scenarios, kanban is often used in projects or situations where the issues and features are required in products that are already advancing through development. It focuses on a flexible approach to development and continuous development periods, meaning that it works well with ongoing projects and projects with everchanging requirement periods. (Planview, 2023)

Case study – Agile methodology in games development

Throughout this case study I will aim to discuss the Agile project management methodology and how the project methodology, which is Agile in this scenario, impacts the testing method, software product, requirements, and the team members in the project. The purpose of this is further expanded to evaluate the effect of a development methodology and how that impacts the aforementioned aspects of testing, etc.

As mentioned above, the Agile project management methodology is a flexible, fast paced project management methodology and this impacts how the software testing is performed at all levels. Agile works through iterations, and that means that throughout the constant development of the project, it will be accompanied by testing and therefore changes made to the project to accommodate the test results.

This case study is to focus on evaluating the impacts of choosing Agile for this project, and the specific impact and implications that has for software testing both in this project and in general.

Advantages and disadvantages of Agile

As mentioned above, Agile provides many benefits accompanied by their fair share of downsides, and this will be discussed below.

Advantages include:

- Flexibility in development
- Changes made to accommodate the results of testing and development
- Faster feedback loops allowing for early detection and resolution of defects
- Better for everchanging requirements as it allows for flexibility
- Enhanced visibility into project progress, allowing for more beneficial and accurate decision making to take place
- More opportunities for different software builds and releases, meaning that there is less time until the product is created and released
- Ability to adapt to not only changing requirements, but also changing technologies and practices

Disadvantages include:

- A steeper learning curve for members inexperienced with Agile development due to the need to understand the core principles and practices of Agile
- A potential for scope creep and lack of project boundaries which can lead to more complications

- High dependencies on communication and close collaboration, something that might not work in distant geographically placed members
- Reliance on the individual members to work effectively
- Limited formal procedures and little documentation
- Inability to provide project timelines and costs due to the everchanging nature of Agile, leading to planning and budgeting issues
- Not suited for strict deadlines and clear requirements

Overall, this places Agile as a project management methodology that should not be used when there are clear outlines of what must be completed, alongside strict deadlines that cannot be modified. This means that although it might not work well for a project like a website, something like a game that is constantly evolving over the course of development would be much more suited to Agile as there are constant changes made to the project based on newly discovered limitations and bugs of the game engine and game itself.

Game development with Agile

Using the Agile project management methodology, I focused on making and developing a 3D puzzle game and this presented a number of interesting situations that I had to focus on while developing. The game in this scenario had a number of requirements, such as collision detection, lives, score, and maintaining an 8+ minimum age range. Although these requirements were not complex, they lead to me having to develop my game in a specific way and this impacted the final outcome of the game.

The Agile methodology was implemented by outlining all of the things that I would have to include in the game, such as multiple levels, and the puzzles that the player had to solve, and from there I organised them into the order that I would do them. This meant that one week I would work on doing one feature, and the next week I would move onto the next and therefore I would be able to progress smoothly, provided that there were no issues. In regard to testing, I specifically made sure to introduce testing at the end of the week to ensure that there were no faults with the code and gameplay features that I had introduced, and this lead to me spending my week in the following format:

- Monday: Fixing any previous issues discovered on Thursday and Friday
- Tuesday: Continuing Monday or moving onto introducing new features
- Wednesday: Continuing Tuesday
- Thursday: Finishing feature implementation and light testing
- Friday: Completing testing and making notes for Monday

This shows clearly how the Agile project methodology specifically impacted the testing of my project, as I had to do it at specific times to allow for development to complete successfully, and then the testing would occur alongside it at the end.

Overall, this lead to the positive impact of producing a polished final game that allowed for a successful project and a satisfactory result. The improved responsiveness of Agile development allowed for instant feedback based on the features I introduced, alongside the swifter detection and resolution of any defects that occurred in my game, and if I was to run this project again I would use Agile.

One of the largest challenges for using Agile in this project was the inability to work on features that might require long term development or just features that would benefit from being worked on all at once rather than on a week-by-week basis, and this was overcome by working on these specific features on the side week through week and producing notes to ensure that I did not lose track of development.

Alternative solution

If I found that Agile was an inadequate project management methodology for developing the game, or I wished to try another methodology, I would choose to use Waterfall, as it has a focus on a clear development of game features and would be the complete opposite of Agile development, leading to a different final product of the project.

Additionally, Waterfall would be an interesting alternative because the requirements of the project were clearly outlined from the start and therefore it would not require any further involvement from the client, making it suitable for Waterfall development.

Conclusion

To conclude, I found that Agile was a very successful project management methodology for the games development project. It worked well with the fluidity and flexibility of making a game and proves that this kind of project management methodology is well suited for something like making a game.

Choosing Agile definitely made an impact on the testing methods of the game, as testing had to be conducted constantly throughout development and therefore I was able to experiment with the different kinds of testing available, like unit testing, acceptance testing, etc. If I had chosen another kind of project management methodology, I do not believe that I would have been able to test as extensively and with as much freedom as I did with Agile, meaning that the project management methodology clearly significantly impacts the testing methods involved with projects.

I have made an extensive effort to ensure that this case study is structured logically, while using both technical engineering and software development terms accurately while reducing the amount of "jargon" in the case study, and this could be easily read by someone like an apprentice software tester.

Case study – Waterfall methodology in website development

Throughout this case study, I will discuss the Waterfall project management methodology and how it impacts the testing method, software product, requirements, and team members in the project. The purpose is to evaluate the effect of the Waterfall development methodology on these aspects of testing and more.

As mentioned above, the Waterfall project management methodology is a structured and strict approach to project management, and this impacts how software testing is performed at different levels. Unlike Agile, which works through iterations, Waterfall follows a sequential approach where each phase is completed before moving to the next. This inflexibility in the development process can impact the testing activities, as changes made to the project to accommodate test results may require significant effort and time-consuming modifications.

This case study aims to evaluate the impacts of choosing the Waterfall project management methodology for this project, and the specific impact and implications it has for software testing, both in this project and in general

Advantages and disadvantages of Waterfall

As mentioned above, Waterfall provides many benefits accompanied by their fair share of downsides, and this will be discussed below.

Advantages include:

- Waterfall follows a sequential approach where each phase of the project has to be completed before moving onto the next, ensuring that there is constant progression based on completion
- Waterfall needs clear and well-defined requirements prior to the project beginning as this crucial for the project management methodology.
- Waterfall also ensures that there is a dedicated focus on each phase of development by enabling thorough planning, design, development, and testing
- Waterfall ensures that there is a high focus on documentation and writing out everything that is taking place during the project
- Although it seems like a negative, the inflexibility of Waterfall can also be an advantage in some projects that focus on minimal change and stability.
- It is well suited for large scale projects that often have well defined requirements and clear roadmaps for what has to be done

Disadvantages include:

- The most notable disadvantage is the limited flexibility of Waterfall, as changes are often detrimental to the project at later stages
- There is a high risk of scope creep due to the lack of accommodation for changes in Waterfall
- Late identification of issues can often have devastating consequences for the project when using Waterfall, meaning that issues and risks that are not accommodated for are brutal
- There is a limited customer involvement in Waterfall as the only time when stakeholders are involved, is at the beginning during the requirements planning for the project
- A reduced team collaboration often occurs with Waterfall as well, due to the fact that work is assigned between members and there can be limited need for communication when everything is predetermined

Overall, Waterfall may be more suitable as a project management methodology when there are clear outlines of what must be completed and strict deadlines that cannot be modified. Waterfall follows a sequential approach with limited flexibility for changes, making it less adaptable to evolving requirements or unforeseen issues. However, for projects like a website that require structured and strict planning, Waterfall may be more appropriate. On the other hand, for projects like a game that are constantly evolving over the course of development, Agile may be better suited, as it allows for frequent changes based on newly discovered limitations and bugs of the game engine and game itself.

Web development with Waterfall

Using the Waterfall methodology, I focused on making a website to fulfil the client requirements that were clearly outlined from the beginning. The requirements from the beginning included subjects like a minimum of 5 pages, an image carousel, a video, and a form. These were not complex requirements however given that this was my first time making a website, there were many hurdles that I had to get through in order to successfully achieve this.

The Waterfall project management methodology was implemented into the project by focusing on taking a sequential approach to development, including clearly separating the distinctive phases of development, such as requirements planning, design, development, testing and finally deployment. I worked on each of these different phases in sequential order, starting from requirements planning and only progressing onto the next phase once I had successfully completed it entirely. In regard to testing, due to the fact that Waterfall follows this sequential order, it had to occur right at the end of my project and this was drastically different to developing in Agile.

Waterfall only allowing testing at the end heavily influenced the final product as any bugs or issues I found with my website during the testing phase were time consuming to fix and led to a lot more time being invested in the project than initially accommodated for. Additionally, the fact that testing is only completed at the end meant that some features, although functional, did not operate as intended and ultimately lowered the quality of the final product. I am sure that if I were to work in a team with experienced users of Waterfall and web development, some of these issues would not occur in the testing phase, however given that this was my first time working with both, unfortunately these issues occurred.

Overall, the Waterfall project management methodology lead to me being able to successfully create a website, including being able to accurately test the website once completed. The constant documentation of Waterfall and strict control over the changes to requirements enabled me to constantly move forward with the project.

The largest challenge that I had to overcome with the Waterfall project management methodology was the inflexibility to do things like testing and modifications to the requirements whenever I thought it would be required. In some cases, this is most definitely the strength of Waterfall, however due to my inexperience I do not believe that I was fully able to utilise all of the benefits it offers.

Alternative solution

If I had discovered that the Waterfall management methodology was unsuited for web development, I would have chosen to attempt it using the Agile project management methodology. This is due to my experience with Agile from previous projects and the ability to make changes when needed, including testing constantly to ensure that all of my feature’s work effectively when all meshed together at the end.

Additionally, I believe that the final product of the project would be distinct when comparted to this project as the core principles of Agile, including the ability to test constantly, are the complete opposite of Waterfall and therefore make an interesting final product.

Conclusion

To conclude, I found that Waterfall was a mildly successful project management methodology for making a website. While it was effective at ensuring that the project continued to progress forward and had lots of documentation to allow for reviews in the future, I am ultimately not satisfied with the final product, and I think that I would have had more success with other project management methodologies. That being said, I do believe that Waterfall is well suited for making websites, however my inexperience with Waterfall and web development was what dragged the project down.

Choosing Waterfall definitely had an intense impact on the testing methods in my website’s development. This is largely due to the fact that the testing all had to occur at once right at the end of the project. The inflexibility of Waterfall meant that all of this had to be done all together at the end, and this lead to me not performing testing as extensively and thoroughly as I would have liked, mostly due to the amount of testing that had to be done meaning that I was rushing through it. I do believe that if I was more experienced with Waterfall then I would have been able to use a wider variety of testing methods in the project and it would have been to a level I was satisfied with, however currently I did not like testing with Waterfall and the lack of depth involved with the testing plays a large part of that.

I have taken great care to structure this case study logically, using accurate technical engineering and software development terminology while minimizing the use of jargon. My aim was to make the case study easily understandable, even for someone like an apprentice software tester.

Bibliography

Chart, L. (2022, October 6). *Waterfall project management methodology*. Retrieved from Lucid Chart: https://www.lucidchart.com/blog/waterfall-project-management-methodology

Gillis, A. S. (2023, April 19). *Acceptance testing*. Retrieved from Tech Target: https://www.techtarget.com/searchsoftwarequality/definition/acceptance-test#:~:text=Acceptance%20testing%20is%20a%20quality,testing%20or%20end%2Duser%20testing.

Katalon. (2023, April 23). *What is regression testing*. Retrieved from Katalan: https://katalon.com/resources-center/blog/regression-testing

Microfocus. (2023, April 22). *Functional testing*. Retrieved from Micro Focus: https://www.microfocus.com/en-us/what-is/functional-testing#:~:text=Functional%20testing%20is%20a%20type,with%20the%20end%20user's%20expectations.

Planview. (2023, April 22). *How to use Kanban in software testing*. Retrieved from Planview: https://www.planview.com/resources/articles/kanban-software-testing/#:~:text=At%20the%20simplest%20level%2C%20Kanban,and%20what%20work%20is%20complete.

Tech Target. (2023, April 22). *System testing* . Retrieved from Tech Target: https://www.techtarget.com/searchsoftwarequality/definition/system-testing#:~:text=System%20testing%2C%20also%20referred%20to,full%2C%20integrated%20system%20or%20application.

TechTarget Contributor. (2023, April 19). *Unit Testing*. Retrieved from Tech Target: https://www.techtarget.com/searchsoftwarequality/definition/unit-testing#:~:text=Unit%20testing%20is%20a%20software,independently%20scrutinized%20for%20proper%20operation.

Usability gov. (2023, April 22). *Usability testing: How To*. Retrieved from Usability Gov: https://www.usability.gov/how-to-and-tools/methods/usability-testing.html#:~:text=Usability%20testing%20refers%20to%20evaluating,watch%2C%20listen%20and%20takes%20notes.

Wrike. (2022, October 6). *What is Agile*. Retrieved from Wrike: https://www.wrike.com/project-management-guide/faq/what-is-agile-methodology-in-project-management/